Interactions en 3D : cycle de vie du geste, de la génération à sa consommation

Mohamed-Ikbel Boulabiar

THÈSE / Télécom Bretagne
sous le sceau de l'Université européenne de Bretagne

pour obtenir le grade de Docteur de Télécom Bretagne
En accréditation conjointe avec l'Ecole Doctorale Sicma
Mention : Sciences et Technologies de l'Information et de la Communication

présentée par
Mohamed-Ikbel Boulabiar
préparée dans le département Logique des usages, sciences sociales & sciences de l'information
Laboratoire Labsticc

Interactions en 3D : Cycle de vie du geste, de la génération à sa consommation

Thèse soutenue le 01 juin 2015
Devant le jury composé de :

Jean-Yves Antoine
Professeur, Université de Tours / président

Géry Casiez
Professeur, INRIA - Université de Lille 1 / rapporteur

Nadine Couture
Professeur, Ecole Supérieure des Technologies Industrielles Avancées (ESTIA 2)- Bidart / rapporteur

Emmanuel Dubois
Professeur, IRIT - Université de Toulouse / examinateur

Thierry Duval
Professeur, Télécom Bretagne / examinateur

Gilles Coppin
Professeur, Télécom Bretagne / co-directeur de thèse

Franck Poirier
Professeur, Université Bretagne-Sud / directeur de thèse

N° d'ordre : 2015telb0355

Sous le sceau de l'Université européenne de Bretagne

Télécom Bretagne

En accréditation conjointe avec l'Ecole Doctorale Sicma

Interactions en 3D : Cycle de vie du geste, de la génération à sa consommation

---

Thèse de Doctorat

Mention : Sciences et Technologies de l'Information et de la Communication

Présentée par **Mohamed Ikbel Boulabiar**

Département : Logique des Usages, Sciences Sociales et de l'Information (LUSSI)

Laboratoire : LabSTICC Pôle : CID

Directeur de thèse : Franck Poirier, Professeur, Université de Bretagne Sud

Soutenue le 01 Juin 2015

Jury :

M.	Jean-Yves Antoine,	Professeur,	Université de Tours (Président)
M.	Géry Casiez,	Professeur,	Université de Lille 1 (Rapporteur)
Mme.	Nadine Couture,	Professeur,	ESTIA (Rapporteur)
M.	Emmanuel Dubois,	Professeur,	Université de Toulouse (Examinateur)
M.	Thierry Duval,	Professeur,	Télécom Bretagne (Examinateur)
M.	Gilles Coppin,	Professeur,	Télécom Bretagne (Directeur de thèse)
M.	Franck Poirier,	Professeur,	Université Bretagne Sud (Directeur de thèse)

Remerciement

Ce mémoire de thèse représente le travail de longues années et qui a inclus plusieurs passages très difficiles. La thèse n'aurait jamais pu aboutir sans l'aide et l'encouragement de plusieurs personnes dans mon entourage. Je tiens d'abord à exprimer toute ma reconnaissance à mes directeurs Gilles Coppin et Franck Poirier. Leurs patience, aide et encadrement au cours de cette thèse ont poussé vers sa réussite surtout pendant les périodes d'exploration et de doutes.

Je remercie aussi Thierry Duval pour ses corrections et remarques pertinentes qui m'ont aidé vers la fin de la thèse. Je remercie Olivier Grisvard pour les réunions et les propositions de développements. Je remercie les rapporteurs et les membres du jury qui ont accepté de lire mon manuscrit et évaluer mon travail tout en insistant sur la correction de certains éléments pour assurer la qualité de la thèse.

Je remercie mes collègues de bureau Mathieu Simonnet et Arnaud Grancher pour leurs encouragements, conseils et critiques constructives qui m'ont aidé à orienter la thèse et résoudre plusieurs problèmes. Je remercie Ameni Ben Letaifa pour l'encouragement et le suivi pendant la période de rédaction ainsi que les membres de ma famille pour le support inconditionnel que j'ai eu au cours de mes études ainsi qu'à leurs encouragement perpétuel

tout au long de mes études.

Je remercie finalement Baptiste Leroy, Benoit Rouxel, Chiara Nistico, Emile Verdurand, Iyas Alloush, Julie Soulas, Magnus Larsson, Romain Picot-Clemente, Soazig Perrigault et Pierre-Yves Pillain pour leurs questions, aide, corrections d'articles, amitié et bon accompagnement dans le laboratoire et pour avoir participé dans les tests des algorithmes.

À tous ces intervenants, je présente mes remerciements, mon respect et toute ma gratitude.

Résumé

Dans le domaine de la reconnaissance gestuelle, les mouvements humains sont observés, reconnues et transformés en primitives fonctionnelles pour contrôler un système ou manipuler un objet. Les chercheurs en Interaction Homme-Machine ont plus particulièrement étudié le suivi des gestes réalisés après le contact avec l'objet à manipuler (geste apparent). Nous montrons dans cette thèse qu'interagir et manipuler un objet 3D est un processus plus large qui commence par la vision, qui inclut le rapprochement, la saisie, la manipulation et qui se termine avec la "consommation des événements" dans les applications cibles. Dans cette thèse, nous avons collecté et organisé un état de l'art sur l'interaction Homme-machine provenant de différents points de vue (vision, neuropsychologie, de saisie et techniques). Nous avons créé un système qui suit les mouvements des utilisateurs sur une table mais également au dessus de la surface à partir d'un nuage de points 3D. Nous avons spécifié des cas d'activités gestuelles et nous les avons utilisés dans deux applications. Nous avons aussi proposé une nouvelle façon de créer des applications adaptables aux nouvelles formes d'interactions en se basant sur un bus logiciel.

mots clés : geste, posture, saisie, ihm, cycle, reconnaissance, 3D, bus logiciel

Abstract

In the gesture recognition domain, human movements are tracked, recognized and mapped to functional primitives to control a system or to manipulate an object. Human-Computer Interaction researchers have particularly focused on the tracking of the gesture made after the contact with the object to be manipulated (apparent gesture). We show in this thesis that interacting and manipulating a 3D object is a wider process that starts with vision, includes reaching, grasping, manipulating and ends with "event consumption" in target applications. In this thesis, we have collected and organized a HCI literature review coming from many fields (vision, neuropsychology, grasping and technical). We have created a system that tracks, from low level 3D point-cloud input, user movements on the table but also above the surface. We have specified multiple cases of gesture activities and we used them in two applications. We have also proposed a novel way of creating future adaptable applications to new forms of interactions using a software bus.

keywords : gesture, posture, grasping, hci, cycle, recognition, 3D, software bus

Table des matières

Table des matières **vi**

Table des figures **xii**

1 Introduction **1**

1.1 Contexte de la thèse 1
1.2 Objectif de la recherche 2
1.3 Motivation et énoncé du problème 3
1.4 Contributions 4
1.5 Organisation du manuscrit 5

I Théories sur le geste **7**

2 Taxonomies du geste **8**

2.1 Introduction 8
2.2 Évolution des définitions 8
2.3 Continuums du geste 9
2.3.1 La main, communication en double sens 9
2.3.2 Gestes dans un discours 9
2.3.3 Relation Geste/Discours dans le cerveau 12
2.4 Classifications gestuelles 13
2.4.1 Taxonomie de Karam 13
2.4.2 Taxonomie de Scoditti 15
2.4.3 Taxonomie de Baglioni 16

2.4.4 Taxonomie de Choi 18
2.5 Techniques de notation des gestes 18
2.6 Correspondances geste-action 21
2.7 Conclusion et synthèse 21

3 Recensement des sources du geste 23
3.1 Introduction 23
3.2 L'apraxie gestuelle 24
3.2.1 Les troubles moteurs et cognitifs 25
3.2.2 Liaison entre le trouble et le symptôme 25
3.2.3 Modèle du geste de Signoret et North 26
3.3 Vision des objets dans l'espace 26
3.3.1 Les voies visuelles 27
3.3.2 Vision des éléments 29
3.3.3 Vision de la 3D 30
3.3.4 Visualisation du corps 32
3.4 Les voies motrices par rapport à la vision 33
3.5 La proprioception 35
3.5.1 L'illusion de pinocchio 35
3.5.2 L'illusion des mains croisés 36
3.5.3 La place des illusions dans l'interaction 36
3.6 La zone d'interaction 37
3.7 Les propriétés de la main 38
3.7.1 Évolution de la main 38
3.7.2 Morphologie de la main 39
3.8 La saisie des objets 40
3.8.1 Types de Saisie 45
3.8.2 Plans d'opposition 46
3.9 Conclusion 46

II Technologies relatives au geste 47

4 Les technologies de reconnaissance du geste 48

4.1 Introduction 48
4.2 Les périphériques, les algorithmes et la main 48
4.3 Les périphériques à détection mécanique ou à base de MEMS 49
4.3.1 Le Microphone 49
4.3.2 La Wiimote 50
4.3.3 Le GameTrak 51
4.3.4 Le Razer Hydra 51
4.3.5 La Myo 52
4.3.6 Génération des tracés 52
4.4 Les périphériques à base de caméras 53
4.4.1 La Sony Playstation Eye 53
4.4.2 Les caméras modifiées 54
4.4.3 La Leapmotion 55
4.4.4 Le module de vision PiCam 56
4.4.5 La Kinect 1, projection en lumière structurée 57
4.4.6 Les caméras à temps de vol 57
4.4.7 Les caméras infrarouges professionnelles 59
4.4.8 Les caméras de détection de chaleur 60
4.5 Synthèse des périphériques de capture 61
4.5.1 Évolution des performances et de la précision 61
4.5.2 Zones de capture 62
4.5.3 Coûts 62
4.6 Conclusion 63

5 Algorithmes de reconnaissance 64

5.1 Introduction 64
5.2 Algorithmes de reconnaissance des tracés 65
5.2.1 Les algorithmes en Soft Computing 65
5.2.2 Les algorithmes géométriques 65

5.3 Geste multitactile 70
5.4 Les algorithmes de reconnaissance en imagerie 74
5.4.1 La détection du flux optique 74
5.4.2 Détection de la peau humaine 75
5.4.3 Utilisation des gants et des patchs colorés 75
5.4.4 Amplification des critères invisibles 76
5.4.5 Recalage d'un nuage de points 77
5.4.6 Critères de comparaison 78
5.5 L'utilisation des bibliothèques de simulation physique 78
5.6 Conclusion 79

III Contributions et développements 80

6 Extension du domaine gestuel 81
6.1 Introduction 81
6.2 Définitions du geste et de la manipulation 82
6.3 Les éléments d'une interaction gestuelle 82
6.4 Naturalité et apprentissage 85
6.5 Positionnement par rapport aux études gestuelles 86
6.6 Simplification de la taxonomie des gestes de manipulation . 87
6.7 Prédiction des manipulations gestuelles 89
6.8 Les choix des technologies de capture 90
6.9 Conclusion 91

7 Système de reconnaissance développé 92
7.1 Introduction 92
7.2 Description du système et des choix 92
7.3 Système de suivi 94
7.3.1 Gestion du flux de la Kinect 94
7.3.2 Algorithme de marquage des blobs 96
7.3.3 Algorithme de suivi des blobs 97
7.4 Système de détection de la main 99

7.5 Conclusion 101

8 Reconnaître les activités de la main 102

8.1 Introduction 102
8.2 Reconnaissance des postures 103
8.3 Reconnaissance des gestes simples 104
8.4 Enregistrement des informations 105
8.5 Pointage sur table 107
8.6 Les autres modules de reconnaissance 109
8.6.1 Les couches d'interaction 109
8.6.2 La détection des clics 109
8.6.3 La reconnaissance des classes de base des gestes . . . 111
8.6.4 Le suivi de plusieurs mains/utilisateurs 111
8.7 Pointage dans l'espace 113
8.8 Reconnaissance des gestes complets 115
8.9 Prédictibilité des gestes 115
8.10 Évolution avec un autre capteur 115
8.11 Conclusion 116

9 La Manipulation 3D dans la pratique 117

9.1 Introduction 117
9.2 Application des gestes 3D dans un contexte 2D 118
9.2.1 Architecture et choix techniques de l'application . . . 118
9.2.2 Scénario de test 120
9.3 Application des gestes 3D dans un contexte 3D 120
9.3.1 Architecture de l'application 121
9.3.2 Scénario de test 122
9.4 Critiques et visions d'améliorations 123
9.4.1 La notion de couches d'interactions 123
9.4.2 Le déclenchement des modules de reconnaissance . . 124
9.4.3 Le prototype représenté en couches applicatives . . . 124
9.4.4 Occultation par la main 125

9.5 Conclusion 126

10 Conclusion **127**

A Au delà du geste, les applications interactives **130**

A.1 Introduction 130

A.2 Applications en tant que canevas 131

A.2.1 Définition d'un bus logiciel 131

A.2.2 Anciens travaux liés 132

A.2.3 Réduire les applications à leurs canevas 133

A.3 Scénarios 134

A.3.1 Scripter des applications graphiques 134

A.3.2 Programmation interactive 135

A.3.3 Factorisation du développement d'applications 136

A.3.4 Compositeur d'applications 138

A.3.5 Documentation interactive 140

A.4 Conclusion et limites 141

B Publications **142**

Bibliographie **143**

Table des figures

2.1 Continuité du geste sur l'échelle du discours 10
2.2 Relation du geste avec les propriétés linguistiques. 11
2.3 Exemple d'un pantomime tirée de l'article [Xu et al., 2009], l'opérateur simule l'ouverture d'une boite. 12
2.4 Les zones communes d'activation pour les gestes symboliques et un langage parlé. Tirée du même article [Xu et al., 2009] 12
2.5 L'acteur dans l'expérience [Andric et al., 2013] 13
2.6 Visualisation de la taxonomie des gestes selon Karam, on voit les 4 catégories de base 14
2.7 L'espace de classification de Scoditti des interactions gestuelles basées sur les accéléromètres 16
2.8 Espace de conception des périphériques d'entrée de Mackinlay . 17
2.9 Taxonomie des interaction gestuelle sur mobile de Baglioni . . . 17
2.10 Les éléments d'une interaction gestuelle 3D selon [Choi et al., 2014] 18
2.11 Le codage du geste par McNeill selon sa zone de génération . . . 19
2.12 Le modèle de notation des gestes de Choi 20
2.13 Exemple de gestes codifiés par le modèle de Choi 20
2.14 Une technique de codification de la posture et du geste de la main [Baudel and Beaudouin-Lafon, 1993] 20

3.1 Notre vision pour la redéfinition du geste pour maîtriser la chaîne de production gestuelle 23

3.2 Avec un objet et selon le contexte, il faut choisir le bon gestème ainsi que le réaliser suivant une bonne sérialisation de kinèmes. [Sève-Ferrieu, 2005] 27
3.3 La version standard de l'illusion avec deux cercles centraux physiquement identiques montré en figure (a), et la version où le disque à droite est plus large physiquement mais semble identique par illusion 29
3.4 Les espaces 3D pour définir les positions et pour définir les objets. [Breuker, 2013] 32
3.5 Modèle sensoriel du corps humain par Wilder Penfield [1] 33
3.6 Le système de vision guide la saisie des objets en proposant une saisie stable à gauche, et instable à droite chez une partie des patients [Goodale and Milner, 1992] 34
3.7 L'espace d'interaction en prenant le corps comme centre et repère de l'orientation [Sève-Ferrieu, 2005]. 37
3.8 L'espace d'interaction continue 2D/3D de Marquardt 38
3.9 La saisie parfaite entre le pouce et l'index 39
3.10 Les méthodes d'interactions présentés dans plusieurs vidéos futuristes tirée du blog de Bret Victor. 40
3.11 Les méthodes d'interactions naturelles avec les objets mêlant la saisie de ces objets par Bret Victor. 41
3.12 La relation de l'humain avec un outil ou un instrument par Bret Victor. 41
3.13 Les gestes de manipulation de la main selon Bullock 42
3.14 La classification de Bullock des manipulations avec saisie d'objet selon l'axe de la main 43
3.15 La taxonomie de la saisie incluant la forme de l'objet reprise depuis [Cutkosky, 1989] 44
3.16 Les facteurs influençant une saisie [Wimmer, 2011]. 45

4.1 Les périphériques jouant le rôle d'un filtre aux informations de la main 49

4.2 La wiimote, le module MotionPlus, et le Nunchuck 50
4.3 La Razer Hydra 51
4.4 La Myo par Thalmic Labs 52
4.5 La caméra de la playstation 3, PS3 Eye 53
4.6 La caméra stéréoscopique du Playstation 4 54
4.7 Le capteur PS3Eye modifié et personnalisé en ajoutant un lentille télescopique 55
4.8 Le capteur Leapmotion 55
4.9 Le module de capture PiCam de Pelican Imaging 56
4.10 La première version de la Kinect, et la visualisation du motif infrarouge émis et utilisé pour la capture 3D 58
4.11 La deuxième version de la Kinect (temps-de-vol) 58
4.12 Les caméras Intel RealSense 59
4.13 Le module de capture FLIR 60
4.14 Les zones de captures de chacun des périphériques offrant une sortie 3D 62

5.1 Extraction des caractéristiques géométriques d'un tracé gestuelle selon la thèse de Rubine 66
5.2 Visualisation des gestes possibles avant la fin du geste actuel . . 66
5.3 Première étape de normalisation de l'algorithme 1$ 67
5.4 La rotation du geste vers un angle initial de l'algorithme 1$ . . 67
5.5 La correspondance dans le calcul des distances pour l'algorithme $1 dans (a), $N dans (b), et $P dans (c) 69
5.6 Les étapes de traitement d'un tracé pour créer un tableau d'indices normalisés 70
5.7 L'algorithme de quantification des angles ne permet pas de faire la distinction entre un motif, et sa répétition 70
5.8 Machine à états de la reconnaissance des gestes 71
5.9 Les familles de gestes multi-tactiles reconnus 72
5.10 Calcul de la rotation dans le moteur de reconnaissance des gestes tactiles 72

5.11 Le moteur de création et de détection des gestes en les considérant comme des expression régulières Proton 73
5.12 Photo montrant l'utilisation du flux optique pour suivre la main [2] 74
5.13 Extraction de la peau humaine depuis une caméra et génération du contour 75
5.14 Utilisant des gants colorés par Wang pour la détection de la posture de la main 76

6.1 Le cycle d'une interaction gestuelle 84
6.2 L'impact de la posture initiale de la main sur le geste final à réaliser, l'utilisateur évite la rotation dans le sens anti-horaire et utilisation. 87
6.3 Les familles principales de gestes 3D 88
6.4 Les gestes mutli-tactiles représentés avec des objets 3D par Gabriele Meldaikyte 88
6.5 La prédiction de la manipulation gestuelle 90

7.1 L'installation de la Kinect 93
7.2 Traitement du flux de données reçus depuis la Kinect 95
7.3 Les quatre étapes d'étiquetage des blobs 96
7.4 Calcul de la distance entre les boites englobantes de deux images, dans 2 cas différents [Senior et al., 2006] 98
7.5 Visualisation du blob contenant la main et le bras 98
7.6 Étapes pour la détection de la main 99
7.7 entrée et sortie du module de détection de la main 100
7.8 Détection du doigt pointeur et de la main 101

8.1 Mécanisme de reconnaissance de la posture de la main en utilisant le contour 103
8.2 Les 4 gestes 3D testés 105
8.3 Le nuage de point des main dans la scène 106
8.4 Le nuage de point de la main 106
8.5 Pointage sur la table avec la main 107

8.6 Visualisation du point de pointage sur la table 109
8.7 Différentiation de la signification selon la couche d'altitude de manipulation 110
8.8 Détection du clic dans la couche d'interaction la plus basse . . . 110
8.9 Les 4 classes gestuelles à reconnaître 111
8.10 Gestion de la reconnaissance multi-main et multi-utilisateur . . 112
8.11 Étape de calibrage pour la sélection des limites du mur 113
8.12 La détection de l'intersection entre la main et le mur 114

9.1 Architecture de l'application 2D consommant les gestes 118
9.2 L'interface de l'application 2D développée 119
9.3 La projection de l'interface 3D sur la table 121
9.4 Architecture de l'application 3D consommant les gestes 122
9.5 Les couches d'interaction autour d'un objet 123
9.6 Le système de reconnaissance représenté en hiérarchie de modules 125
9.7 L'occlusion par la position de la main 126

10.1 L'utilisation des gestes dans la voiture sans nécessiter de voir l'interface 128

A.1 Connection des événements de rotation de la main pour contrôler les mouvements des carrés dans inkscape, aucune modification du code de l'application n'est fait. 133
A.2 Implémentation du dessin interactive de figures géométriques. L'utilisateur peut sélectionner une valeur et visualiser le changement de façon instantané dans Inkscape pendant qu'il bouge le curseur. 135
A.3 Implémentation du concept des "magic lenses". L'application complètement séparés a une fenêtre semi-transparente et peut modifier la couleur des éléments qui se situent derrière son cadre dans Inkscape 136

A.4 Concept du développement d'une application multi-périphérique où l'application fournit un canevas de dessin et exporte ses fonctionnalités dans le bus 139

Chapitre 1

Introduction

1.1 Contexte de la thèse

La vision est l'un des sens, peut être le plus important, à partir desquelles les humains perçoivent le monde. La lumière réfléchie depuis les objets et reçue par l'œil, permet de générer des signaux électriques transmis par la rétine vers les différentes parties du cerveau. En présence d'objets à manipuler, l'être humain construit une représentation cognitive de la scène, et planifie ou adapte une stratégie d'interaction avec les objets reconnus. Cette interaction caractérisée par un feedback permanent, est possible de l'appliquer à des scènes et des objets virtuels. Via cette interaction l'utilisateur dicte les nouvelles coordonnées de l'objet dans l'espace, ou le changement de l'une de ses propriétés.

Avec l'apparition de nouvelles technologies, ces interactions peuvent être augmentées par le remplacement de l'une de ses parties par des moyens virtuels. En effet, on peut remplacer les objets réels par d'autres qui sont virtuels et que nous pouvons les présenter à l'humain à travers des écrans ou des illusions de 3D stéréoscopiques, holographiques ou augmentées. L'interaction avec ces nouveaux objets virtuels nécessite aussi des moyens de capture et de compréhension des mouvements de l'utilisateur.

Les nouveaux périphériques d'entrée et les nouveaux capteurs trans-

forment les postures et gestes de la main en messages provoquant des transformations géométriques dans l'espace numérique des objets virtualisés.

Cette commande pour gérer un système doit se faire d'une manière efficiente, rapide, précise et concise, mais ce n'est pas toujours le cas aujourd'hui. Avec un peu de recul, la manière avec laquelle on manipule les objets virtuels n'a pas trop changé dans la majorité des cas depuis une dizaine d'année, dans les environnements de programmation, les outils de créativité ou même les cartes tactiques développées dans un environnement virtuel de commande.

L'idée de manipuler un système en utilisant les mains sans presque aucune interface visible où on doit taper, toucher ou déplacer un objet était et est encore un rêve de réalisateurs de films de science fiction. Elle est aussi le travail perpétuel de nombreux chercheurs en Interaction Homme-Machine qui n'arrivent pas encore à remplacer la souris, un périphérique inventé il y a une cinquantaine d'année, par un autre qui est plus efficace ou plus facile à utiliser. Nous souhaitons malgré tout nous inscrire dans cette perspective et nous proposons de créer un système capable de reconnaître le but de l'utilisateur en suivant ses simples gestes et de l'aider à accomplir ses manipulations.

1.2 Objectif de la recherche

Il existe de nombreux travaux de recherche sur la gestion de l'entrée utilisateur et sur la reconnaissance des gestes. Une partie de ces travaux traitent aussi la correspondance gestes aux traitements applicatives et montrent que le problème est assez large et assez complexe à traiter. Un moyen pour améliorer la maîtrise de ce domaine est d'avoir plus de recul et de revoir les choses autrement en essayant de nouvelles pistes. L'objectif principal de cette thèse est d'explorer les travaux de recherche des autres domaines qui s'intéressent aux manipulations de la main et les organiser avec un point de vue qui se concentre sur la manipulation gestuelle et notamment d'avancer

dans la compréhension de la signification d'un geste et sa capture, en essayant de manier les limites du problème. On espère dans notre approche aller jusqu'aux limites des travaux de recherche qui sont liés au geste, même si ces travaux font partie d'autres domaines de recherches que l'Interaction Homme-Machine, ordonner ces travaux suivant un fil conducteur et essayer de voir comment ils interagissent ensemble.

Les travaux de recherche autour du geste vont nous aider à comprendre l'humain et sa manière de générer les gestes, et surtout ceux qui sont en relation avec la manipulation d'objets. On considère que le traitement du geste peut commencer dès la vision d'un objet qui définit sa génération et se termine au delà de la manipulation de l'objet lorsque l'information gestuelle générée est traitée à l'intérieur d'une application informatique consommatrice de ces événements où ils finissent leur chemin.

Vu que le chemin commence par l'humain, passe par les périphériques et algorithmes de capture et finit dans une application, il nécessite l'inclusion de plusieurs domaines de recherche. Nous allons essayer d'aller le plus loin possible dans les concepts à maîtriser en développant aussi un moteur de reconnaissance depuis une entrée bas niveau où on maîtrise nos propres algorithmes. Notre objectif est d'avoir le maximum de retour d'expérience en essayant d'élargir les limites du traitement gestuel actuel et de proposer un mécanisme que l'on puisse utiliser par la suite pour continuer les recherches.

1.3 Motivation et énoncé du problème

Dans ce travail, nous remettrons en cause la zone d'étude gestuelle en voyant le problème avec le plus de recul possible. Une telle vision nécessite la redéfinition du geste lui même de façon plus large. On propose donc de revoir la définition du geste et sa signification dans le contexte des travaux de manipulation d'objets virtuels.

La motivation de ce travail réside dans la compréhension de l'aspect humain. Il faut donc

— étudier et cerner les aspects cognitifs
— tester les limites des périphériques de captures et les algorithmes associés
— essayer de maitriser la plus grande partie de la chaine
— suivre le geste dans sa transmission à l'intérieur des applications

Nous creusons majoritairement les aspects cognitifs, on teste les limites des périphériques de captures et des algorithmes tout en essayant de maîtriser la plus grande partie de la chaîne. Nous suivons aussi la transmission du geste dans les applications.

1.4 Contributions

Notre principale contribution dans cette thèse est l'organisation d'une sélection des travaux de recherches en vision, cognition, saisie, et manipulation, autour de la reconnaissance gestuelle. Le but de cette organisation est de surpasser la façon habituelle pour traiter la problématique de la reconnaissance et consommation gestuelle qui concentre jusqu'à maintenant sur une partie spécifique de la relation geste-objet et chercher des nouvelles perspectives.

En plus de cette contribution théorique, on propose une contribution technique visant à maîtriser le cycle de la reconnaissance gestuelle à partir d'une entrée de bas niveau comme le nuage de points 3D reçu par une caméra de profondeur et jusqu'à arriver à générer des messages gestuels à consommer par les applications interactives plus haut dans le cycle.

Ces contributions peuvent ouvrir la voie pour des futures analyses sur la prédiction des saisies et des manipulations d'objets. Le modèle théorique gestuel élargit les frontières d'étude du geste en proposant un continuum remontant dans le temps avant la manipulation apparente avec l'objet, et le système technique permet de supporter le modèle théorique en proposant la plate-forme d'expérimentation pour le suivi de la forme de la main avant la manipulation et pour viser la forme d'approche et de saisie de la main qui

influencent le geste apparent dans le reste du cycle. Ce manuscrit contient aussi une proposition pour la génération d'applications interactives d'une nouvelle manière en facilitant la communication avec un bus logiciel. Cette dernière contribution en particulier permet de créer des applications facilement adaptables aux messages gestuels et aux nouvelles formes d'entrée sans nécessiter une réécriture du code source à chaque fois ou une modification de l'application elle même. Elle nécessite au début une externalisation des fonctionnalités de l'application sur un bus logiciel ce qui donne le contrôle au créateur des événements gestuels.

1.5 Organisation du manuscrit

Le manuscrit est divisé en 3 parties, la première partie est une partie théorique qui regroupe 2 chapitres d'état de l'art. Le premier expose les taxonomies théoriques et habituelles sur le traitement de la reconnaissance gestuelle et surtout sa liaison avec le discours. Le deuxième est celui sur les domaines de recherches autour d e gestes incluant la vision, l'approche, la saisie, et la manipulation projeté sur le geste. Ce chapitre expose les domaines qui tournent autour du geste et qui ont leur place temporelle avant le déroulement du geste sur l'objet.

La deuxième partie du manuscrit est une partie technique contenant aussi 2 chapitres. Le premier focalise sur les périphériques de capture des gestes. Il expose les différents moyens physiques de captures les plus utilisés pour capturer les mouvements humains. Le deuxième chapitre présente les algorithmes employés pour extraire les informations importantes des périphériques de captures et les transformer vers des informations de plus haut niveau. Nous parcourons des anciennes ou des nouvelles méthodes de transformation du flux sortant des périphériques et parfois invisibles à l'œil vers des messages utiles.

La troisième partie regroupe les chapitres avec principalement les contributions effectuées et elle est organisée en quatre chapitres. Cette partie com-

mence par un chapitre pivot, il lie notre analyse de l'état de l'art théorique avec les choix techniques à notre disposition pour valider et argumenter les choix que nous avons fait par la suite. Il propose aussi une manière de simplification pour gérer les gestes et faciliter leurs classification en grandes familles en s'inspirant des gestes tactiles. Le chapitre suivant présente le système de reconnaissance que nous avons développé et installé et les briques algorithmiques de bases permettant de filtrer les sources de données brutes et les préparer pour une reconnaissance gestuelle. Le chapitre qui suit explique les algorithmes utilisés pour la détection des gestes et des postures ainsi que quelques pistes testées. Et le dernier chapitre contient une présentation des systèmes développés et servant de consommateurs des messages gestuels générés.

Bien que l'on a l'impression de couvrir tout le cycle gestuel, il manque des aspects en ingénierie pour l'évolution des systèmes, et pour ceux là, ils sont présentés dans un chapitre à part dans l'annexe. Ce chapitre propose, à travers quelques scénarios, une ré-architecture des applications pour faciliter leurs adaptabilités aux nouvelles sources d'entrée sans une réécriture totale de la logique de l'application.

Première partie

Théories sur le geste

Chapitre 2

Taxonomies du geste

2.1 Introduction

Dans la littérature, il y a eu plusieurs études sur le geste aboutissant à des taxonomies différentes. Selon le domaine de recherche les points de vue divergent sur une définition unique d'un geste. Dans ce chapitre nous présentons une vue générale des études gestuelles les plus marquantes en point de vue IHM. Les études dans ce chapitre focalisent sur les gestes sans nécessiter explicitement la présence d'un objet cible.

2.2 Évolution des définitions

Un geste est défini d'une façon générale par le mouvement d'une partie du corps comme la main ou la tête. Le mouvement peut avoir une signification en lui-même ou associé à une conversation. Dans le cas des mouvements de la main, la configuration des doigts avant, pendant et à la fin du mouvement est importante. La configuration de la main à un instant s'appelle posture, et avec le geste, elle porte une information. Le langage des signes est un exemple où les gestes et les postures de la main ont une signification temporelle et spatialisée.

2.3 Continuums du geste

Les premières études autour du geste comme celle d'Adam Kendon [Kendon, 1972] l'ont analysé en focalisant sur son contexte de création. Les premières taxonomies se sont tournées vers le discours puisque c'est le domaine d'étude le plus évident et le plus accessible à la recherche avant l'abondance des outils informatiques. Dans sa thèse, Adriano Scoditti [Scoditti, 2011] a évoqué le débat existant entre les différents chercheurs sur la façon avec laquelle il faut traiter le geste. Les chercheurs qu'on peut considérer initiateur des deux analyses du gestes sont Paul Ekman et al. [Ekman and Friesen, 1981] et Claude Cadoz [Cadoz, 1994]. Ekman a proposé l'étude du geste comme moyen de communication avec les systèmes élevant le caractère linguistique, alors que Cadoz voyait le geste comme moyen de communication instrumentale, de la même façon qu'un musicien utilise son instrument et donc nécessite la présence de cet instrument.

2.3.1 La main, communication en double sens

Claude Cadoz [Cadoz, 1994] considère le geste comme un canal de communication à double sens, la main à la fois émet et reçoit de l'information. Comme moyen de perception, le toucher, généralement concentré sur les dernières phalanges, permet de renseigner sur la température des objets et la perception tactile qui s'approche de la perception spatiale de l'oeil. Il permet aussi d'accéder aux propriétés proprioceptives, connaître le poids, sa forme et les mouvements des objets manipulés. Mais la main permet aussi de produire un message et une expression qui contient elle de l'information utile, comme la gestique du chef d'orchestre.

2.3.2 Gestes dans un discours

Ekman et Friesen [Ekman and Friesen, 1981] ont organisé les gestes suivant des catégories. La plus importante dans notre cas est celle qui contient

les adaptateurs, renommés ci-après "les manipulateurs" parce qu'ils sont générés en présence d'objets. Cette catégorie nécessite la présence d'un objet et la transformation de ses propriétés spatiales avec la main. Par contre, les "manipulateurs" incluent aussi le corps comme objet manipulable, que ce soit le corps de l'opérateur : "self-adaptor", et le corps d'un autre : "alter-adaptor".

L'un des travaux les plus cités est celui d'Adam Kendon [Kendon, 1972] qui a étudié le geste dans le contexte du discours. Dans sa taxonomie, il a créé des groupes de gestes selon leurs implications dans un discours appelé le continuum de Kendon.

Gesticulation Un mouvement qui contient un sens lié aux discours accompagnant. C'est le type de geste le plus fréquent qu'on trouve quotidiennement. Les mouvements ne sont pas restreints aux mains, mais incluent aussi la position de la tête et les bras. Dans un discours, il n'y a pas une relation directe entre leurs génération et une hésitation ou une pause.

Emblèmes Ce sont les signes conventionnels dans une culture, comme le signe "OK". Kendon les appelle les gestes percutants. L'aspect culturel est très présent dans ces gestes. Ils peuvent être créés en parallèle avec un autre groupe de gestes.

Pantomime Un geste ou une série de gestes ayant un sens pour remplacer une narration. Ils sont générés sans aucun discours.

Signes Ce sont des mots lexicaux, comme le langage des sourds-muets. Ils ont leurs propres convention et structures linguistiques et grammaticales.

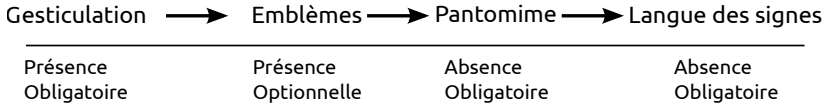


FIGURE 2.1 : Continuité du geste sur l'échelle du discours

McNeill [McNeill, 2005] a repris l'étude de Kendon, en montrant qu'il y a plus d'un seul continuum. McNeill les a augmenté vers ce qu'il appelle continuua.

Dans un autre continuum, McNeill classifie les gestes selon la présence ou l'absence des propriétés caractéristiques linguistiques. Le langage des signes, comme indique la figure 2.2, a ses propres structures grammaticales, alors que la gesticulation n'en contient aucune. C'est pour ceci qu'on les trouve dans les extrémités.

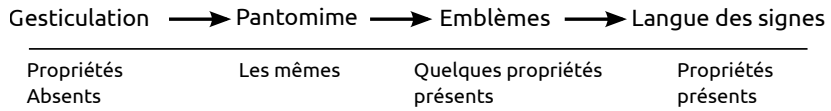


FIGURE 2.2 : Relation du geste avec les propriétés linguistiques.

Beaucoup d'études ne sont pas sorties du cercle de la relation du geste avec le discours et le continuum de Kendon comme le travail de Quek et al. [Quek et al., 2002]. Quek se concentre sur le type de gestes nommé Gesticulation contrairement aux travaux qui ont traité la manipulation et les gestes sémaphoriques. Il constitue l'une des premières liaisons avec le domaine de l'interaction en automatisant le traitement des données collectées via une caméra. Il définit par contre les gestes de manipulation, comme ceux pour contrôler une entité manipulée. Cette manipulation est réalisée via une relation avec le mouvement de la main ou le bras et les gestes sémaphoriques comme les gestes statiques ou dynamiques appartenant à un dictionnaire prédéfini. Dans son travail, il a enregistré avec une caméra puis suivi les mouvements de la main ainsi que le discours. Les mouvements suivants les différents axes sont codés et suivis dans le temps. Une codification simple a été définie et utilisée pour noter l'utilisation d'une seule main "1H" ou de deux "2H".

2.3.3 Relation Geste/Discours dans le cerveau

D'après une étude par Jiang Xu [Xu et al., 2009], les gestes symboliques et le langage parlé sont traités dans la même zone du cerveau. Les analyses sont faites en utilisant le mécanisme fMRI [1] pour détecter les activités neuronales. Les gestes symboliques testés sont les pantomimes et les emblèmes.

FIGURE 2.3 : Exemple d'un pantomime tirée de l'article [Xu et al., 2009], l'opérateur simule l'ouverture d'une boite.

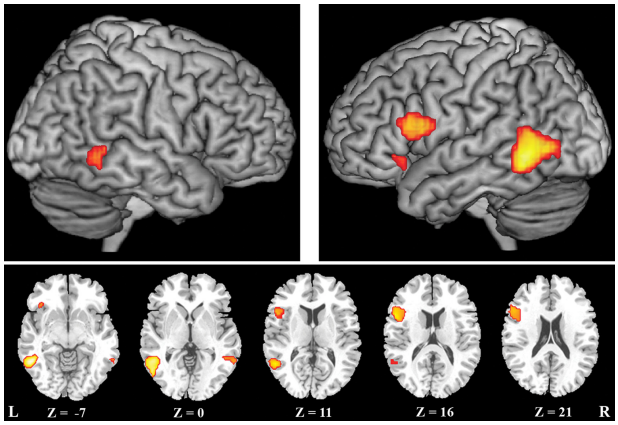


FIGURE 2.4 : Les zones communes d'activation pour les gestes symboliques et un langage parlé. Tirée du même article [Xu et al., 2009]

L'étude de [Andric et al., 2013] lie encore plus le discours parlé, le geste symbolique, et la saisie des objets. Les sujets de l'expérience regardent un acteur en vidéo, figure 2.5, en train de faire des emblèmes, lire un énoncé dans le sens de l'emblème, et saisir un objet.

1. functional Magnetic Resonance Imaging

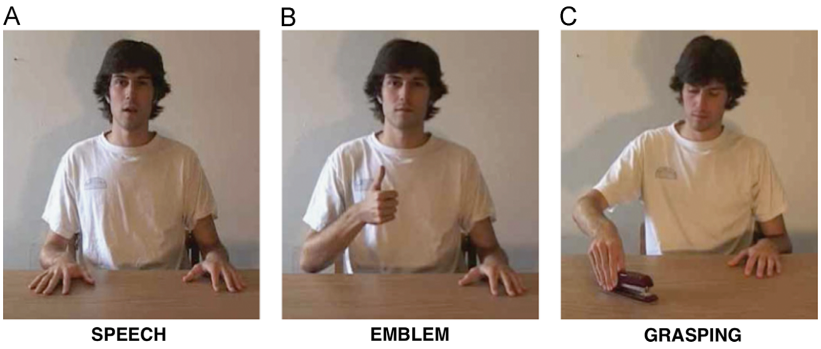


FIGURE 2.5 : L'acteur dans l'expérience [Andric et al., 2013]

Ce traitement par le cerveau, dans la même zone ou dans des zones spécifiques proches, signifie une relation proche entre le geste, le discours parlé et la saisie que ce soit dans l'apprentissage initial, ou dans la génération. Sachant que d'après le travail de Hauk et al. [Hauk et al., 2004], une activité cérébrale captée par les analyses en fMRI se déclenche dans les mêmes régions du cerveau si on pratique une opération, et si on lit un texte contenant une référence à cette même opération. Les références aux membres du corps comme par exemple la main, le bras ou le visage, par la lecture ou par la vision activent ces mêmes régions.

2.4 Classifications gestuelles

2.4.1 Taxonomie de Karam

On présente ici la classification de Karam [Karam and Schraefel, 2005] regroupant les travaux depuis les années 60. Pour bien gérer la diversité et points de vue dans le domaine gestuel, et pour englober les différentes études, Karam propose une hiérarchie commençant par les formes du geste dans un domaine d'application. Dans les domaines il considère les périphériques d'entrée-sortie. Ensuite, il propose 4 catégories principales dans sa taxonomie : style gestuel, domaine d'application, technologie d'entrée et finalement la réponse système ou la sortie.

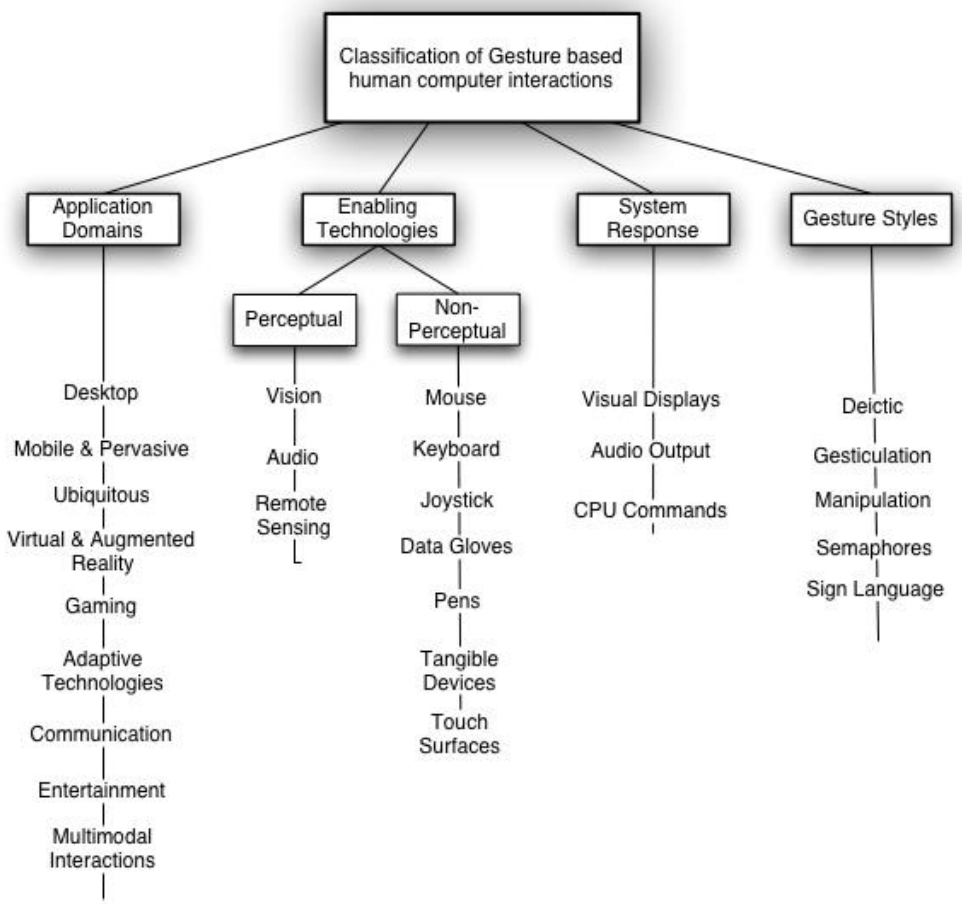


FIGURE 2.6 : Visualisation de la taxonomie des gestes selon Karam, on voit les 4 catégories de base

Karam traite aussi la correspondance entre dans les degrés de libertés de l'entrée et les dimensions du système manipulé. Une correspondance directe est lorsque on manipule des objets 2D sur une surface elle aussi 2D d'une façon similaire à une partie des travaux de Rubine [Rubine, 1992] sur la manipulation directe gestuelle. Les paramètres du geste, comme sa position, sa taille, son orientation, sont mis en correspondance comme paramètres avec la commande. Le travail de Rubine concerne bien sûr des gestes bidimensionnels. a correspondance tient compte de plus de paramètres d'entrée si

les périphériques détectent d'autres propriétés comme la pression.

Karam cite le travail de Hinckley et al. [Hinckley et al., 1998] sur l'utilisation des périphériques tangibles pour manipuler des objets 3D présentés sur l'écran. Hinckley référence plusieurs éléments de l'état de l'art utilisant les deux mains pour finalement les utiliser pour la visualisation des images neurochirurgicales.

La taxonomie de Karam référence les technologies gestuelles mais dans ce manuscrit, les technologies et les algorithmes seront traités dans deux chapitres à part.

2.4.2 Taxonomie de Scoditti

Adriano Scoditti [Scoditti et al., 2011] s'appuie sur un point de vue émergent des travaux sur les périphériques d'entrée, et propose une autre taxonomie pour l'interaction gestuelle. En travaillant sur les accéléromètres, Scoditti propose un espace de classification gestuel présenté par la figure 2.7 où l'abscisse définit la manipulation physique et l'ordonnée correspond à l'ancienne liste des tâches de James D. Foley des années 80.

Cette méthodologie de classification est similaire à celle de Mackinlay et al. [Mackinlay et al., 1990] et elle est représentée par la figure 2.8. Cette dernière figure présente le type du mouvement (Linéaire/Rotationnel) suivant les axes (X,Y,Z), la propriété qui change (Mouvement linéaire : Position, Force, Mouvement de rotation : Rotation, Torsion), ainsi que si le changement est absolue ou relatif par rapport à une ancienne valeur. Cette taxonomie a permis de détecter de façon visuelle les zones couvertes par les périphériques d'entrée et ceux qui restent à explorer et à proposer. La manipulation et la communication gestuelles peuvent donc être vues comme une méthode d'entrée une fois étudiées dans le domaine d'IHM. Les périphériques d'entrée standards peuvent être vus comme des filtres des significations des mouvements de la main. Dans son manuscrit de thèse, Scoditti [Scoditti, 2011] présente les différente études gestuelles et propose un tableau de classification où on peut voir encore les classes gestuelles tirés

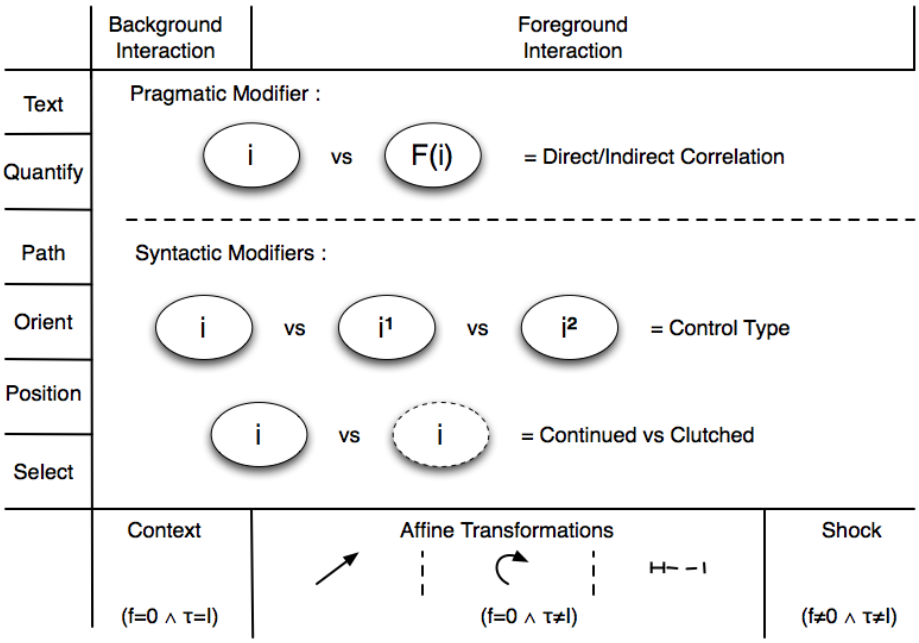


FIGURE 2.7 : L'espace de classification de Scoditti des interactions gestuelles basées sur les accéléromètres

du continuum de Kendon. En utilisant les informations des capteurs d'accélération, il y une focalisation sur les propriétés qu'on peut tirer de ces capteurs pour avoir un geste enrichi. Les propriétés reposent par contre sur l'ancienne classification des tâches de Foley qui peut en quelque sorte être considérée comme obsolète vu que les cas utilisations actuelles ont dépassés largement ces tâches limitées en nombre.

2.4.3 Taxonomie de Baglioni

Mathias Baglioni a présenté une autre taxonomie [Baglioni et al., 2009] qui se focalise sur les téléphones portables s'inspirant aussi à la fois de la classification de Karam et de la conception spatiale de Mackinlay. Dans sa première classification représentée dans la figure 2.9 il a surtout mis en valeur le type de mouvement qu'il soit fluide ou impulsif, et le type de contrôle discret ou continu. Ce type de contrôle définit en quelque sorte les limites temporelles du geste de manipulation et de commande. Dans

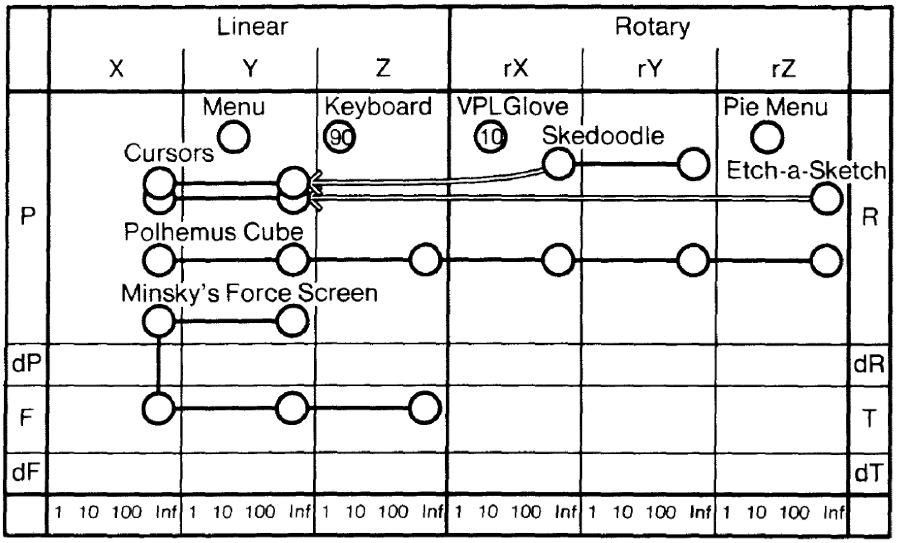


FIGURE 2.8 : Espace de conception des périphériques d'entrée de Mackinlay

la suite de l'article, Baglioni présente une classification sur une matrice de différents travaux gestuels selon le type mouvement, le type de contrôle, le retour d'information, et la nature du geste. L'analyse traite le geste encore d'un point de vue entrée utilisateur via les capteurs du téléphone, sans focaliser sur la main et le positionnement des doigts.

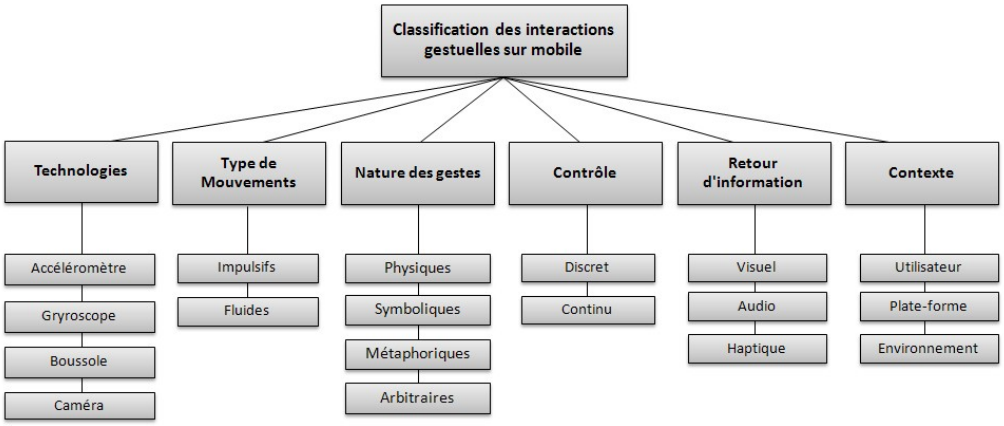


FIGURE 2.9 : Taxonomie des interaction gestuelle sur mobile de Baglioni

2.4.4 Taxonomie de Choi

Une autre étude récente par Choi et al. [Choi et al., 2014] a proposé une hiérarchie des éléments caractérisant une interaction gestuelle 3D avec la main. La figure 2.10 représente ces éléments figurant dans une telle interaction. La taxonomie a pour but de pouvoir noter un geste en le codifiant dans une chaîne de caractère.

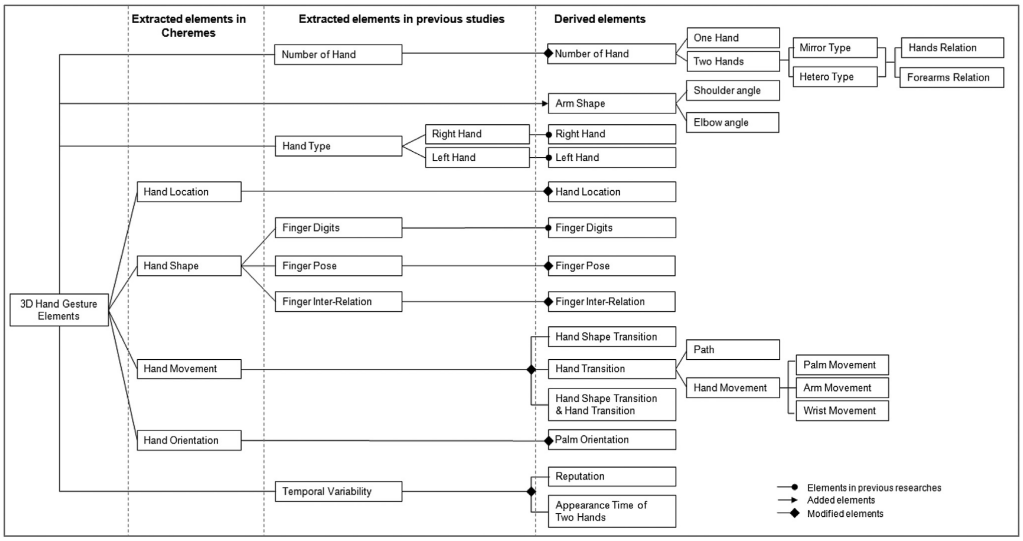


FIGURE 2.10 : Les éléments d'une interaction gestuelle 3D selon [Choi et al., 2014]

2.5 Techniques de notation des gestes

Les études théoriques sur le geste et les taxonomies sont utilisées pour codifier un geste avec une méthode de notation utilisant les symboles alphanumériques. Les notations sont nécessaires si on veut sauvegarder les gestes dans un format plus exploitable qu'une simple vidéo. Ils sont aussi utilisés dans un contexte plus technique pour créer des événements qu'on peut transférer entre un mécanisme de reconnaissance et une application qui interprète et consomme ces événements.

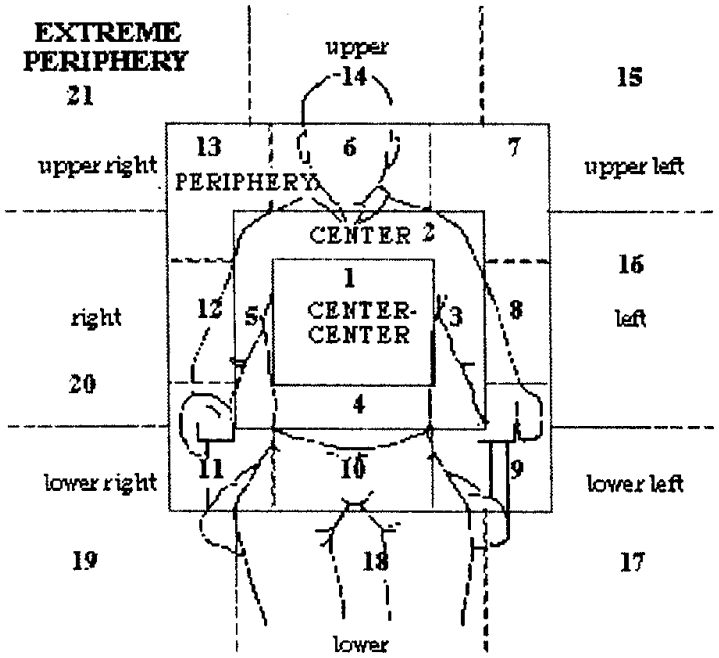


FIGURE 2.11 : Le codage du geste par McNeill selon sa zone de génération

McNeill dans la figure 2.11 propose un mécanisme simple pour coder un geste en se basant sur l'emplacement où il est généré. Comme le montre la figure 2.11 il découpe l'espace autour du sujet dans des zones, et note un geste effectué dans une zone par un nombre. On trouve aussi LibStroke [2] codifiant les mouvements de la souris en des commandes. Cette même technique a été reprise par Choi et al. [Choi et al., 2014] tout en ajoutant les éléments gestuels de la figure 2.10 pour noter les propriétés des gestes effectués dans une zone précise. Ce dernier divise l'espace dans 9 zones seulement par comparaison à McNeill et arrive aussi à spécifier si un geste utilise une ou deux mains.

La figure 2.12 montre le modèle de notation de Choi qui transforme une information mécanique en une chaîne de nombres. La figure 2.13 donne des exemple de ces suites de nombres pour quelques types de posture de la main et du bras.

2. http ://etla.net/libstroke/

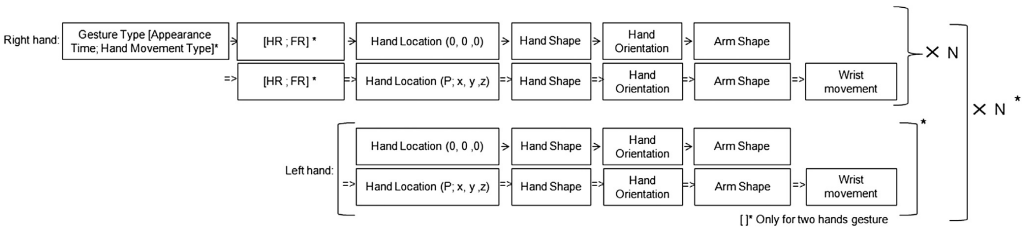


FIGURE 2.12 : Le modèle de notation des gestes de Choi

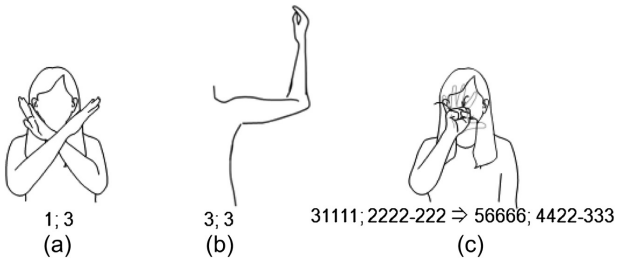


FIGURE 2.13 : Exemple de gestes codifiés par le modèle de Choi

Un autre technique de codification des gestes a été proposé par Baudel et al. [Baudel and Beaudouin-Lafon, 1993]. Cette technique permet de codifier en utilisant des graphiques simples la posture et la dynamique du geste dans le cadre d'un outil de présentation.

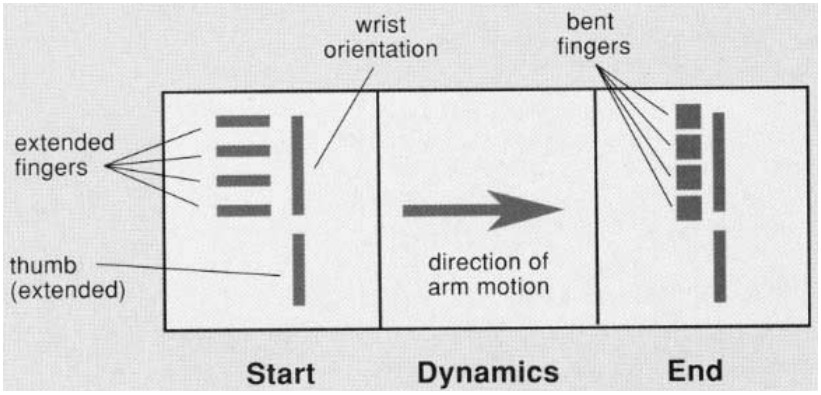


FIGURE 2.14 : Une technique de codification de la posture et du geste de la main [Baudel and Beaudouin-Lafon, 1993]

2.6 Correspondances geste-action

D'un point de vue efficacité et meilleure transmission de l'information gestuelle, il est important de bien lier un geste à une action dans une application. La correspondance peut être facilitée en utilisant des métaphores avec lesquelles les utilisateurs sont habitués.

Dans un cadre de manipulation d'objets 3D visualisés à travers un écran bidimensionnel, il est possible de faire une multitude de gestes avec la main. Dans le travail de Martinet et al. [Martinet et al., 2010], on explore différentes techniques de positionnement en utilisant des surfaces multi-tactiles. En terme d'efficience, les résultats ont montré qu'il n'y a pas une amélioration dans le temps minimal pour accomplir la tâche, mais par contre il y un choix de préférence qui émerge vers la technique de positionnement avec les interactions sur l'écran tactile.

Un autre travail de Reisman et al. [Reisman et al., 2009] utilise les gestes de rotation, redimensionnement, et translation (RST) pour manipuler des objets ou une scène 3D à travers un écran multi-tactile. Dans l'article, ils explorent les différents gestes qu'il est possible d'utiliser pour la manipulation 3D à travers l'écran en résolvant les ambiguïtés entre les objectifs de l'utilisateur et la manipulation à réaliser par le système. Le travail peut être classé en une forme de correspondance entre geste 2D et manipulation 3D.

2.7 Conclusion et synthèse

Dans ce chapitre, on a fait un tour rapide sur les taxonomies du gestes les plus connues qui concentrent leurs analyses à partir d'une relation geste/discours tel que celle de McNeill et qui a fait la base de plusieurs études par la suite. La majorité des travaux présentés constituent l'approche classique pour le traitement de la problématique de la classification et reconnaissance gestuelle. On constate que ces travaux, avec leur valeur scientifique importante, n'ont pas focalisé sur les autres travaux en relation avec le geste mais qui font partie des autres domaines de recherches. À part la relation

avec le discours, on n'étudie pas ce qui se fait avant et après la génération du geste. Les aspects cognitifs et d'interprétation visuelle et activation cérébrale n'ont pas été mis en relation dans un cadre IHM. On a pris la décision d'explorer d'autres domaines de recherche qui peuvent aider dans le contexte gestuel pour nous aider à répondre sur l'aspect naturalité du geste et de la manipulation et trouver les éléments qui les lient ensemble.

Chapitre 3

Recensement des sources du geste

3.1 Introduction

Très peu de chercheurs dans le domaine de l'interaction humain-machine ont approfondi l'étude de l'ensemble des moyens humains incluant : la vision, la perception, la saisie et la manipulation des objets comme point de départ pour finalement arriver à modéliser l'humain et définir la manière avec laquelle il produit les gestes. Dans la majorité des cas c'est le chemin inverse qui est suivi pour trouver les bons gestes qu'un opérateur doit réaliser dans un contexte d'interaction gestuelle avec un système. D'autres études traitent simplement une partie du geste étendu comme la manipulation, qu'on va appeler ici "manipulation apparente" ou "geste apparent" conceptualisée dans la figure 3.1.

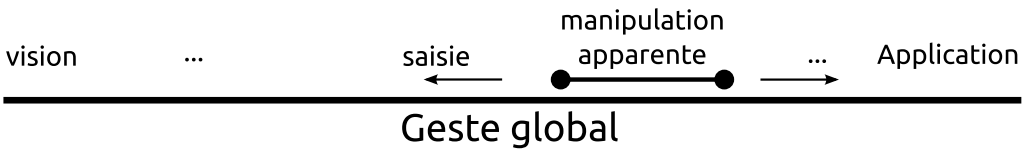


FIGURE 3.1 : Notre vision pour la redéfinition du geste pour maîtriser la chaîne de production gestuelle

Dans ce chapitre, nous commencerons par l'étude de l'humain et sa perception de l'environnement et de lui même, pour arriver finalement à comprendre comment il génère les informations de manipulation des gestes. Les briques d'études n'appartiennent pas forcément au même domaine de recherche et notre but est de rendre les liens entre les différents domaines plus explicites tout en ayant un point de vue général d'un chercheur en IHM.

Nous allons commencer par les troubles gestuels et d'apraxie chez certains patients que nous pensons être l'un des moyens révélateurs sur le cycle de traitement du geste par le cerveau.

3.2 L'apraxie gestuelle

La praxie est un mot grec qui signifie action, et la praxie gestuelle signifie la coordination et l'exécution ordonnée de gestes pour réaliser des mouvements volontaires dont le but final est d'accomplir une tâche. L'activité accomplie est le résultat de l'activité des centres nerveux dans le cerveau.

Par contre, l'apraxie gestuelle est l'inaptitude à réaliser cette série de mouvements, tout en ayant des fonctions motrices et sensitives de base qui sont saines.

Nicole Sève-Ferrieu dans son livre [Sève-Ferrieu, 2005] cite le travail de Luria [Luria et al., 1978] où il assimile une fonction à une activité complexe d'adaptation de plusieurs mécanismes du corps. L'exemple d'activité de pliage d'une feuille en 4 pour l'introduire dans une enveloppe n'est donc pas une opération unique, mais c'est le résultat d'une synthèse entre plusieurs systèmes du corps. Luria a utilisé d'autres termes que l'apraxie, comme l'apractognosie qui est l'absence des réactions aux douleurs et résultante de la perte des "synthèses visuo-spatiales".

La démarche pour déchiffrer ces apraxies, est d'étudier le cycle en entier, et d'analyser les troubles des patients apraxiques.

3.2.1 Les troubles moteurs et cognitifs

Les troubles moteurs et cognitifs sont des moyens pour détecter les symptômes d'un problème dans la génération du geste. Chercher à résoudre un problème à partir de la seule analyse de ses symptômes n'est pas toujours très efficace si on a pas une grande partie des détails, puisqu'une migraine peut être la conséquence de plusieurs causes, et que peut être seule une correction de vue avec une paire de lunettes peut résoudre le problème.

Nicole Sève-Ferrieu [Sève-Ferrieu, 2005] a étudié dans ses travaux les différents troubles qui peuvent arriver chez un patient. Puis elle les a classifiés selon différentes catégories. Elle a évoqué l'apraxie constructive qui signifie l'inaptitude d'un patient à manipuler des éléments dans l'espace. L'expérience invoquée a pour but la construction de quelques dessins sur une feuille, de façon plane ou en perspective.

Les troubles mnésiques sont liés à la mauvaise capacité "d'orientation préalable dans la sélection du matériel". Un autre trouble est l'incapacité d'utiliser le corps comme référence pour l'espace entourant. Avec ce trouble le patient devient incapable d'évaluer les distances entre les objets, entre lui et les objets ainsi que connaître sa zone atteignable ou son étendu corporel.

La reconnaissance visuelle est en effet la première étape du processus de reconnaissance d'un objet. Elle est citée dans le livre comme pouvant être affectée par des troubles. Un humain devrait bien reconnaître les objets devant lui, qu'à défaut, les éléments vus deviennent non significatifs pour déclencher la manipulation adéquate.

3.2.2 Liaison entre le trouble et le symptôme

Reconnaître les troubles n'est pas suffisant pour comprendre le fonctionnement de la chaîne de perception-action, il faut pouvoir comprendre les liens troubles-conséquences. Selon Sève-Ferrieu, il faut analyser cinq niveaux de traitements perceptif : mnésique, linguistique, corporel, visuel et de programmation constituant les maillons d'une chaîne. Il faut les analy-

ser non seulement chez les patients, mais aussi chez les humains sains. Ceci permettra de comprendre le dysfonctionnement et de lier les blocs entre eux. L'exemple le plus simple est l'analyse de la programmation du geste, permettant de voir si la planification des gestes est bien cohérente et organisée. Cette analyse consiste à tester si un humain arrive à comprendre la forme et la fonction d'un objet, sélectionne le bon geste, et programme les micro-mouvements pour produire la manipulation de l'objet.

3.2.3 Modèle du geste de Signoret et North

Les travaux de Signoret et North [Sève-Ferrieu, 2005], considèrent deux niveaux fonctionnels distincts du geste. Après une étude sur les patients ayant des apraxies, il a été découvert que les patients ont l'un de deux problèmes suivants : soit ils ne peuvent pas choisir le bon geste pour saisir un objet, soit ils ont un problème dans la réalisation de la liste des mini-mouvements à effectuer pour arriver à l'objet. Il a donc conclu qu'il y a deux niveaux dans la réalisation du geste. Ces niveaux différencient le geste symbolique tel qu'imaginé, et la réalisation motrice. La production du geste nécessite à la fois le choix du bon gestème et la séquences des kinèmes effectués l'un après l'autre. Un gestème définit le modèle du bon geste à choisir avec un objet, et le kinème est un partie du mouvement à effectuer.

D'après les travaux de Signoret et North, et les études précédentes, on est en face d'une analyse gestuelle de plusieurs niveaux et de plusieurs étapes de construction. La première étape commence par la vision des objets.

3.3 Vision des objets dans l'espace

L'oeil est l'un des organes les plus complexes du corps humain. Il a suivi un processus d'évolution en commençant par des petites cellules qui captent la lumière jusqu'à arriver à un multi-mécanisme composé d'une lentille, une rétine, un iris. L'évolution n'est pas survenue seulement dans l'œil mais aussi dans le cerveau. Le chemin suivi a permis à l'être humain

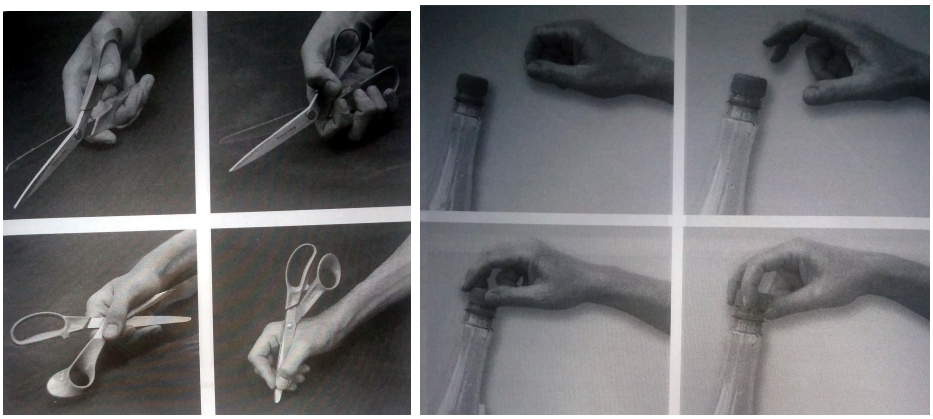


FIGURE 3.2 : Avec un objet et selon le contexte, il faut choisir le bon gestème ainsi que le réaliser suivant une bonne sérialisation de kinèmes. [Sève-Ferrieu, 2005]

de comprendre la troisième dimension dans les objets, même si c'est d'une façon rigide, câblée et facile à induire en erreur dans la plupart des cas.

3.3.1 Les voies visuelles

Le potentiel électrique des neurones change et le voltage chute.

3.3.1.1 Les informations visuelles dans le cerveau

Les informations visuelles de l'œil parcourent le nerf optique pour passer à travers le corps genouillé latéral dans le Thalamus, ensuite vers le cortex visuel où les informations sont organisées sous forme de carte visuelle. La position d'une information dans la rétine est préservée et enregistrée dans des cartes dans le cerveau. Les informations les plus importantes préservées sont les contours et les bords. L'information est enregistrée encore sous une forme chimique et électrique utilisant des neuro-transmetteurs. Les angles d'un objet sont codés en nombre de pointes, plus l'angle est vertical, plus il y a de pointes électriques transmises (spikes).

Ces deux mécanismes bas niveau sont cités pour montrer que la vision n'est qu'une réaction chimique simple avec un système qu'on peut modéliser

avec une carte. C'est le nombre de neurones qui permet de le rendre efficace. Vu comment l'information visuelle est enregistrée dans le cerveau, il peut y avoir des imperfections. Si un humain regarde des informations avec des contours incomplets, le système de neuro-transmissions n'arrive pas à spécifier l'incomplétude, et arrive à activer les neurones qui sont dans la région. Ceci explique comment les lois de visions de Gestalt fonctionnent. Les lois de complétion et continuité sont causées par les neurones du cerveau qui s'activent si les neurones voisins sont actifs. L'information est organisée sous forme d'une carte dans le cerveau. Le caractère spatial de l'enregistrement de l'information dans le cerveau et le fait que des neurones peuvent influencer les neurones proches fait que le système complète les zones sans informations dans une image.

3.3.1.2 La vision et l'action

Dans les travaux de Aglioti et al. [Aglioti et al., 1995], ils ont traité le problème des illusions et leurs relation avec les actions menées avec la main. Il a préparé une expérience avec une version modifié de l'illusion de Titchener comme montre la figure 3.3 montrant deux cercles avec un qui semble plus grand que l'autre. Il a ensuite demandé au participant de déplacer l'un des cercles et il a trouvé que l'illusion n'a pas perturbé la manipulation de l'objet. Il a ensuite conclu qu'il y a deux voies de vision indépendantes dans le cerveau, mais qui s'interagissent.

Dans le travail de Bruno et Bernardis [Bruno and Bernardis, 2002] on trouve une autre expérience mêlant la perception dans le contexte de l'illusion optique de compression de Kanizsa et la manipulation. Sans être inconsistante avec les anciens travaux, cette expérience a montré une nouvelle dissociation entre les jugements visuels et les actions guidées par la vision. Ils ont aussi reporté qu'un effet d'illusion créé dans le système visuomoteur en boucle fermée et en relation avec l'objet, mais pas dans celui en boucle ouverte utilisant un geste de mime. Depuis ces études, on voit que la vision et l'action sont étroitement liés dans le cerveau, et dont des chan-

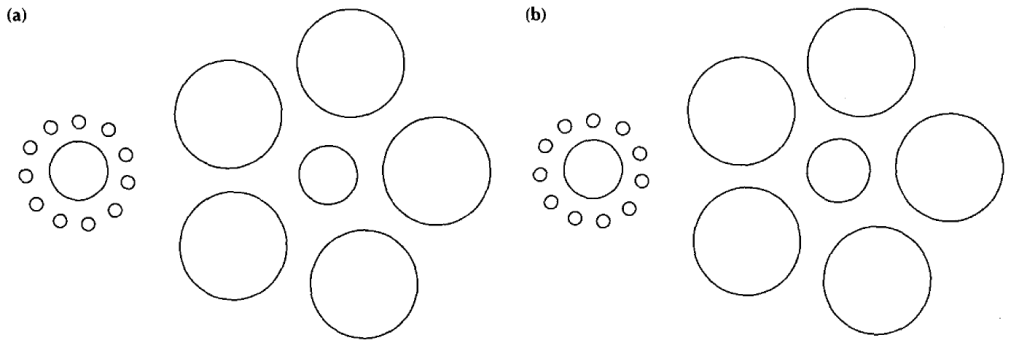


FIGURE 3.3 : La version standard de l'illusion avec deux cercles centraux physiquement identiques montré en figure (a), et la version où le disque à droite est plus large physiquement mais semble identique par illusion

gements dans la manipulation d'un objet réel avec ou sans illusions, peut faire beaucoup de différences.

3.3.2 Vision des éléments

Notre vision des objets et des éléments ne distingue pas un élément seul mais le considère dans sa relation avec son entourage. La Gestalt explique le mécanisme de la perception visuelle par quelques principes et lois simples. Ces principes permettent de compléter un manque d'information, de détecter une anomalie rapidement, ou de détecter la partie intéressante dans une image et focaliser dessus.

Proximité Les objets qui sont proches entre eux sont vus plus liés que ceux qui sont plus loins.

Similarité Les objets qui se ressemblent sont vus comme faisant partie d'un seul groupe.

Continuité La vision préfère voir les objets comme continus et non pas divisés. L'œil suit la forme d'une courbe ou une ligne et évite les interruptions.

Complétion On voit les lignes pointillés comme continues. Notre vision complète le vide entre les morceaux de lignes.

Séparation Figure-Fond Lorsqu'il n'y a pas une limitation claire dans une image, on peut visualiser deux choses dans la même image. L'état bistable est le résultat de l'impossibilité du cerveau de choisir entre ce qui définit le fond et ce qui définit la figure.

Taille relative Les objets qui ont la même taille sont groupés visuellement de la même manière.

Groupement uniforme Le placement des éléments à l'intérieur d'un autre élément ou bien la liaison entre des éléments différents dans les schémas permet d'avoir un groupement visuel.

Espace Blanc Actif Si l'espace vide qui est laissé est petit, il devient un élément actif dans la visualisation et influence la vision.

Destin Commun Les objets qui bougent dans la même direction appartiennent au même groupe. (en anglais elle est nommé "common fate")

Point Focal La vision ne peut se concentrer que sur une zone limitée.

Expérience Précédente C'est l'un des principes les moins forts. Une expérience visuelle précédente influence la vision des nouveaux objets.

Au cours de l'évolution animale, ces principes de vision sont apparu pour permettre l'analyse rapide de l'information surtout en cas de danger et permettre la survie de l'espèce. Il n'est pas nécessaire de voir un prédateur en entier pour s'enfuir.

3.3.3 Vision de la 3D

Le cerveau a plus d'un mécanisme pour comprendre la 3D. Avoir 2 yeux permet d'utiliser l'angle de convergence entre eux, ce qui permet de savoir si un objet est proche ou loin. Un autre cas est la stéréovision qui permet de comprendre l'emplacement des objets en ayant des projections avec un espacement sur la rétine. Les objets qui sont à la même distance dans le monde réel sont projetés avec la même distance sur la rétine. Un objet éloigné a une distance différente.

Avec un seul œil il est aussi possible de voir en 3D. L'occultation d'un objet permet de savoir si un objet est devant un autre ou pas. Le brouillard, la brume, la taille des objets, la convergence des lignes parallèles permettent la même chose. L'être humain peut aussi avoir une idée sur la forme grâce à l'ombre. Il y a une manière fixe de penser que la lumière vient toujours du haut. On voit toujours la forme d'un objet nuancé bombé s'il est brillant du haut plus foncé du bas, et on voit le même objet concave si on inverse la nuance. Le focus de l'œil sur un objet permet d'avoir une idée sur la distance. La lentille de l'œil est plus bombée pour focaliser sur un objet proche. La forme de la lentille est contrôlée par les muscles et cette information permet d'avoir une idée sur la 3D après une opération de fusion d'information dans le cerveau. Le parallaxe de mouvement est lui aussi une autre façon de comprendre la 3D des objets. Les objets les plus proches bougent plus rapidement lorsqu'on se déplace que ceux qui sont loins.

Dans le travail de Joost Breuker [Breuker, 2013], dont le sujet principal est l'acquisition de la connaissance, il parle dans la quatrième partie sur les aspects cognitifs de l'acquisition des informations cartographiques. D'un point de vue beaucoup plus abstrait et haut niveau, il indique que l'être humain acquiert l'information 3D sous forme de 2D+1D. Ou de façon plus explicite on voit l'espace du plan, ensuite les objets qui se sont mis dessus dans une dimension à part. En s'inspirant de travail de Jacob et al. [Jacob et al., 1994] autour de la gestion de l'entrée, on a une une intégralité dans la vision du plan, et une séparabilité avec les objets qui se posent dessus. On voit la 3ème dimension de ce plan comme des objets qui s'empilent l'un sur l'autre.

Breuker pense que la dimension verticale est définie par la gravité. Il pense aussi qu'il y a deux espaces de définition de la 3D, l'un pour définir les positions, et l'autre pour les objets comme le montre la figure 3.4. Cette façon avec laquelle on visualise les objets nous mène à gérer l'espace comme un continuum 2D-3D. L'espace 2D est l'espace de référence qui s'étend par la suite en 3D. Ce continuum sera discuté dans le chapitre 6.6 ayant le but

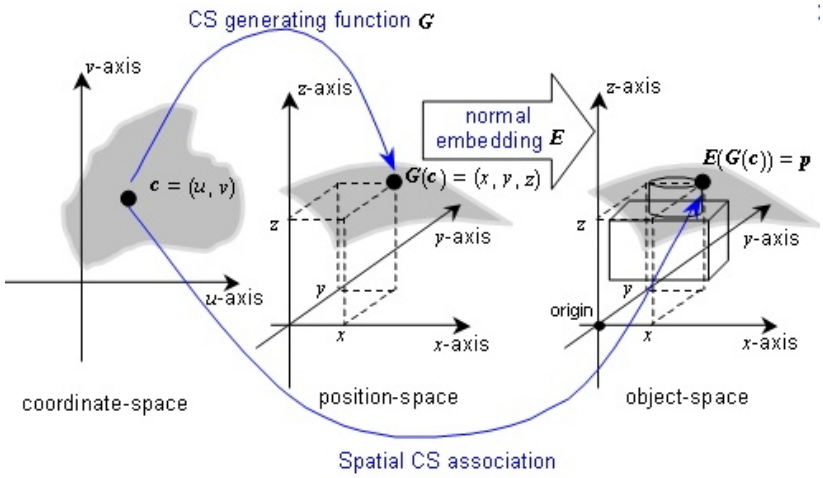


FIGURE 3.4 : Les espaces 3D pour définir les positions et pour définir les objets. [Breuker, 2013]

de simplifier les taxonomies gestuelles.

3.3.4 Visualisation du corps

Pour arriver à contrôler les différents organes, le cerveau maintient un modèle du corps humain. Ce modèle se calibre et se développe durant la période de l'enfance. Durant ses premières années, l'être humain essaye de construire le modèle de son propre corps. Parfois l'apprentissage se fait par imitation des gestes que l'enfant observe chez ses parents. Le modèle sensoriel peut être recalibré dans les zones cibles de chaque organe à l'âge adulte dans une expérience où on porte des miroirs montrant tout l'environnement inversé. Après une période de 2 semaines continues, les sujets de cette expérience arrivent à se comporter naturellement avec leur vision inversé de l'environnement. Une deuxième période de calibrage est nécessaire après l'enlèvement des miroirs.

Il tient compte des informations du déplacement en temps réel. Ces informations là contiennent beaucoup de bruit. Le cerveau arrive quand

1. http ://fr.wikipedia.org/wiki/Homonculus_sensitif

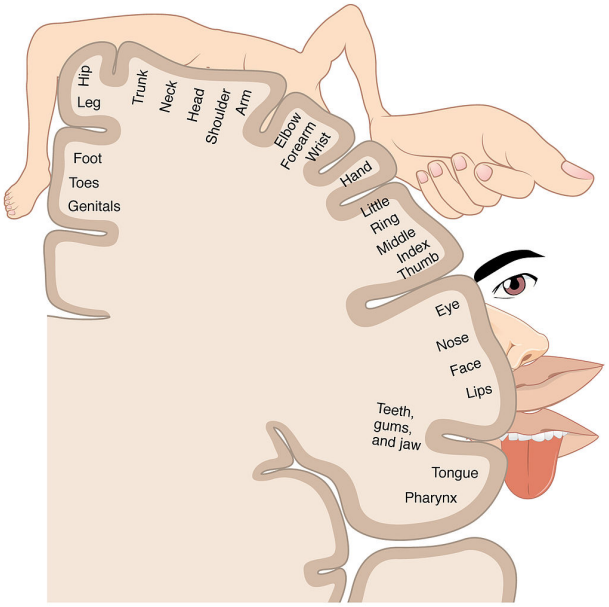


FIGURE 3.5 : Modèle sensoriel du corps humain par Wilder Penfield [1]

même à les supprimer et à faire les déplacements qui permettent d'avoir le minimum d'efforts et le moins de muscles à activer.

3.4 Les voies motrices par rapport à la vision

Dans les analyses précédentes de la vision on a vu qu'il y a deux voies séparées mais interagissant ensemble pour guider la manipulation des objets. Le dixième chapitre du livre de Goldenstein [Goldstein, 2008] traite les systèmes de vision pour l'action et pour la perception. Il suggère que deux systèmes séparés mais qui interagissent ont évolué chez l'humain. L'un pour la perception des objets et l'autre pour gérer les action sur ces objets. Il mentionne aussi que le système visuel traite des informations qui n'ont rien à voir avec la vision comme la synchronisation des rythmes circadiens. Dans le contexte de la vision pour l'action, il cite les travaux sur les rongeurs qui ont montré qu'il existe des modules pré-câblés pour différents

comportements comme l'orientation de la tête ou pour éviter une barrière contrairement aux primates qui ont des systèmes plus flexibles. Bien que les anciens systèmes peuvent encore exister, ils sont modulés par des systèmes de contrôle dans le cortex cérébral. Dans le contexte de la vision pour la perception, il indique que le contrôle flexible visio-moteur permet au primate d'identifier les objets pour comprendre leurs signification et appliquer des relations causales haut niveau impliquant la mémoire, la sémantique, le raisonnement spatial, la planification et la communication. Cette représentation haut niveau permet de former un but. Le travail de recherche de Goodale et Milner [Goodale and Milner, 1992] propose que la voie ventrale permet de construire la représentation perceptuelle du monde et des objets pour savoir si c'est un fruit, alors que la voie dorsale permet le contrôle visuel des actions dirigé à ces objets et donner les informations comme l'emplacement, l'orientation, la taille, et la forme.

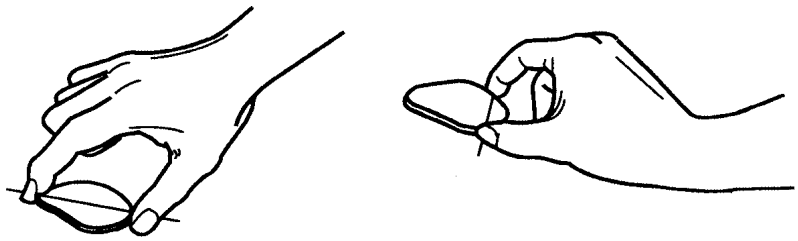


FIGURE 3.6 : Le système de vision guide la saisie des objets en proposant une saisie stable à gauche, et instable à droite chez une partie des patients [Goodale and Milner, 1992]

Goodale a réalisé une expérience représentée par la figure 3.6 sur des patients ayant une ataxie optique et d'autres ayant une agnostique de forme visuelle. L'expérience a montré que les premiers patients n'arrivent pas à saisir les objets de façon stable comme dans la représentation de gauche.

3.5 La proprioception

Le déplacement quelconque d'un segment du corps prévoit d'abord d'avoir un modèle mental initial des différents segments du corps. Il est nécessaire aussi que le cerveau reçoive instantanément les données de chaque muscle pour mettre à jour son modèle mental avec les nouvelles positions. La perception ne s'arrête pas à la connaissance des positions mais va jusqu'à celle des efforts exercés.

Cette connaissance de l'état du corps et du soi à un instant donné est appelée la proprioception. Cette information est traité par le cerveau utilisant la fusion des organes tendineux de Golgi et des fuseaux neuromusculaires. Les organes tendineux sont localisés dans la jonction du muscle et positionnés en série permettant de capturer la force appliquée, alors que les fuseaux neuromusculaires sont localisés dans le muscle et en parallèle permettant de capturer la longueur du muscle. Ces deux systèmes de neurones captent l'étirement et envoient l'information au cerveau suivant des pics répétitifs de potentiels, la fréquence de ces pics est le porteur de l'information.

Mine et al. [Mine et al., 1997], et d'une façon plus haut niveau que les techniques du processus chimique décrites précédemment, ont travaillé sur l'exploitation de la proprioception pour déplacer des objets dans un espace virtuel, de la difficulté de cette tâche sans avoir de retour (feedback) et de la limite dans la transformation des informations de l'entrée utilisateur. Une des solutions pour réduire la difficulté est l'utilisation d'interactions remises à l'échelle du corps humain et utilisant la propriété de l'accès à l'espace de l'objet (reaching).

3.5.1 L'illusion de pinocchio

Dans le livre de [Haggard and Flanagan, 1996], dans le chapitre 17, Lynette Jones détaille les recherches autour de la proprioception. L'expérience de Lackner [Lackner, 1988] sur l'illusion appelée "vibratory myesthetic illusion" montre les limites du système de proprioception du cerveau. Cette

illusion consiste à appliquer des vibrations sur le muscle du biceps pour imiter l'information d'un étirage, le sujet doit aussi en même temps toucher son nez. Il est demandé aussi que le sujet soit mis dans une salle noire ou qu'il ferme ses yeux pour que l'information de vision n'intervienne pas dans l'expérience.

Lorsque le cerveau reçoit l'information émulée de l'étirement et en même temps celle que la main touche le nez, il essaye de corréler les deux et la seule façon qu'il trouve est de la traiter comme un étirement ressenti du nez. L'expérience est aussi appelée l'illusion de Pinocchio.

3.5.2 L'illusion des mains croisés

Le cerveau combine les informations du toucher, de la position du corps et des yeux, ainsi que de la vision. Les yeux dans leur état de fonctionnement ne peuvent pas se déplacer de façon souple, ils s'orientent de façon brusque avec des mouvements qui sont appelés "saccades". Dans l'expérience de Grohe et al. [Groh and Sparks, 1996] le sujet commence par avoir les mains dans une position standard, il bouge sa main droite et essaye de regarder sa main droite à partir d'un point de focus vers le haut. Ensuite il croise ses mains et essaye de refaire la même expérience. Dans le cas normal, les yeux se déplacent directement vers le bon endroit, mais dans le second cas, les yeux se déplacent vers l'autre main.

3.5.3 La place des illusions dans l'interaction

Les nombreuses illusions présentées montrent la facilité avec laquelle on peut tromper les mécanismes de traitements et fusion des informations du corps dans le cerveau. Si on comprend bien la manière avec laquelle le cerveau fonctionne, on peut générer encore de nouvelles illusions et les utiliser pour induire le cerveau à comprendre autre chose. Une application de l'un des mécanismes du cerveau est la création de l'effet de la 3D stéréoscopique à travers deux images dirigées vers chaque œil et qui a été largement utilisée.

3.6 La zone d'interaction

Dans son processus de perception de l'espace, l'humain considère plusieurs zones d'interaction. La première est le corps via la proprioception. La deuxième est la zone d'extéroception, qui est la zone extérieure. Cette même zone se divise en plusieurs parties, dont celle située face à l'opérateur, (là où est dirigée sa vision), et celle qui l'entoure. La zone d'interaction est liée elle aussi à la vision et à la mémoire. D'après le cours "The Brain and Space" sur coursera, certains patients n'arrivent plus à identifier les éléments qui se trouvent à leurs gauche s'ils ont des problèmes dans le cortex de droite.

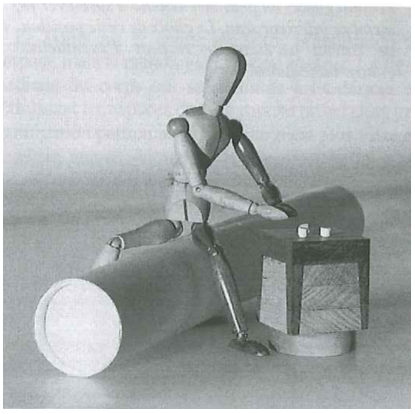


FIGURE 3.7 : L'espace d'interaction en prenant le corps comme centre et repère de l'orientation [Sève-Ferrieu, 2005].

Dans le livre de Sève-Ferrieu [Sève-Ferrieu, 2005] elle fait travailler les patients sur leurs coté négligé par le cerveau pour essayer d'améliorer leurs capacités. Ceci est représenté par la figure 3.7 tirée du livre où le patient travaille dans l'hémiespace négligé avec des contraintes de position.

La zone d'interaction a fait partie de plusieurs travaux de recherches surtout focalisant sur les grandes surfaces tactiles comme celui de Marquardt et al. [Marquardt et al., 2011]. Dans son travail il traite l'espace sur la table et au dessus, représenté par la figure 3.8 comme un espace continu d'interaction qu'il est possible de l'utiliser.

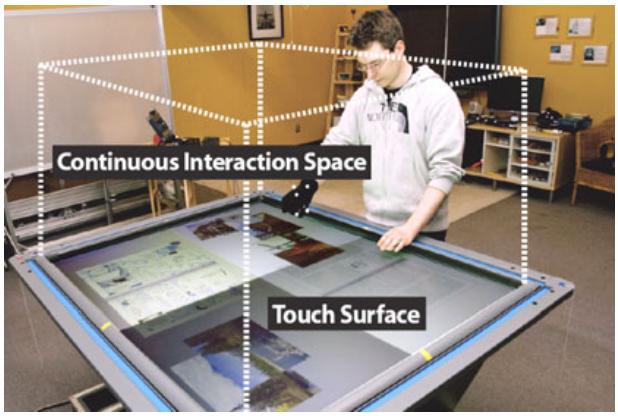


FIGURE 3.8 : L'espace d'interaction continue 2D/3D de Marquardt

3.7 Les propriétés de la main

La main est l'élément terminal dans la manipulation d'objets. Elle a été étudiée pour comprendre sa dextérité de saisie et la reproduire en robotique. Par contre, la main est le résultat de millions d'années d'évolution. Cette évolution a bien sûr eu lieu pour accomplir une fonction spécifique de manipulation. John Napier [Napier and Tuttle, 1993] dans son livre de plus de 150 pages étudie la main et sur les différents aspects, allant de l'évolution jusqu'à la saisie et la pratique des gestes avec ou sans un discours.

3.7.1 Évolution de la main

La main est parmi les organes qui ont eu beaucoup de transformations lors du processus de l'évolution. Elle a assuré plusieurs fonctionnalités en commençant comme un organe latéral pour se déplacer dans l'eau, ensuite transformé pour supporter le poids et se déplacer sur terre. L'utilisation a été ensuite diversifiée suivant les espèces.

Malgré la transformation dans la fonctionnalité, la structure principale de la main reste la même. Le placement des os n'a pas changé et il est facile d'identifier chacun des noms. Chez les primates, la différence est encore moins visible, elle est surtout dans la longueur des doigts et dans l'angle

d'opposition entre le pouce et le reste des doigts. La main humaine permet d'avoir une opposition parfaite entre le pouce et l'index, parfois avec l'intervention du majeur pour améliorer la stabilité. Cette perfection n'est pas atteignable avec la main d'un chimpanzé à cause de la petite taille du pouce.

3.7.2 Morphologie de la main

La main est constituée d'exactement cinq doigts grâce à la présence du même gène régulateur chez les animaux. La main humaine diffère de celle des autres primates par sa capacité à saisir les objets.

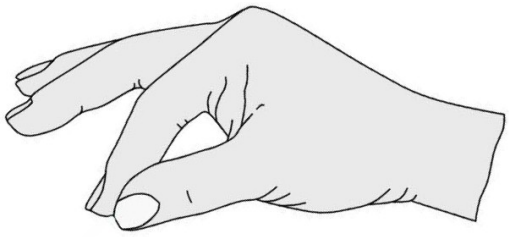


FIGURE 3.9 : La saisie parfaite entre le pouce et l'index

En liaison avec la morphologie de la main, il y a l'étude des degrés de libertés qui sont possibles à utiliser. Le nombre des degrés de libertés est considéré parfois 27, ou 26 [Oikonomidis et al., 2011a]. Le travail Brouet et al [Brouet et al., 2013] essaye de trouver les degrés de libertés qu'une main peut contrôler avec ses 27 degrés de libertés. Dans son travail il indique que vu la morphologie de la main, certains degrés ne sont pas exploités. Les mouvements des doigts sont restreints. L'étude a seulement traité les positions des doigts sur écran tactile et pas dans l'espace. Elle a montré que les utilisateurs impliquent plus de degrés de libertés que ceux nécessaires dans la tâche.

3.8 La saisie des objets

La finalité des mécanismes de vision et de proprioception dans une interaction est de toucher, saisir et manipuler l'objet. On passe ici à l'étape qui précède et détermine la manipulation, c'est la saisie des objets. Plusieurs domaines de recherche étudient la saisie grâce à l'efficience et la dextérité de la saisie humaine. Leurs principal objectif est de chercher comment répliquer la fonction dans les robots et l'automatiser. Le développeur et l'artiste Bret Victor a exprimé en 2011 [2] son reproche aux vidéos présentant le futur comme un domaine d'interaction avec des surfaces de verres avec des touchers utilisant un seul doigt comme montré en figure 3.10.

FIGURE 3.10 : Les méthodes d'interactions présentés dans plusieurs vidéos futuristes tirée du blog de Bret Victor.

Victor a cité le livre de Napier et son article [Napier and Tuttle, 1993, Napier, 1956] sur la façon avec laquelle un être humain manipule les objets. Ces manipulations 3D comme dans la figure 3.11 réalisé par Victor, montre qu'il y a une richesse dans la manipulation qu'il ne faut pas oublier. Il faut améliorer l'interaction pour être plus proche d'une manipulation naturelle avec des objets 3D ordinaires.

Bret Victor a aussi rappelé la définition d'un outil de la même façon tel que présenté dans l'article sur l'interaction instrumentale de Beaudouin-

2. http ://worrydream.com/ABriefRantOnTheFutureOfInteractionDesign/

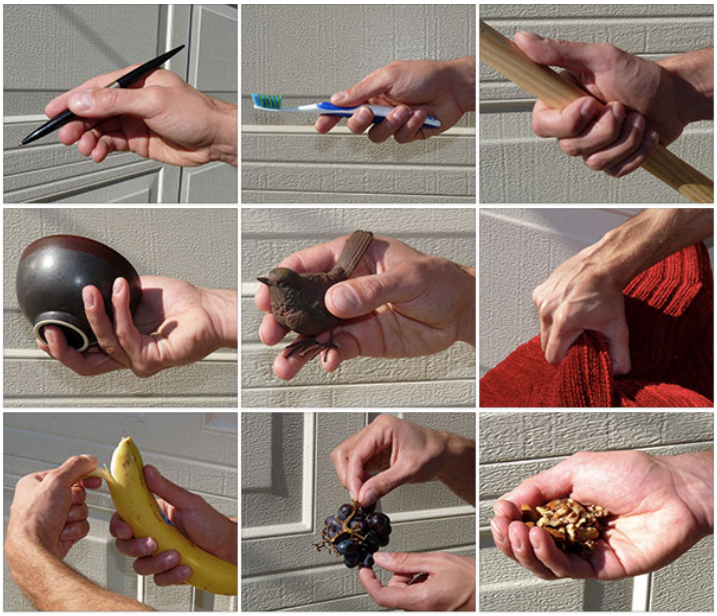


FIGURE 3.11 : Les méthodes d'interactions naturelles avec les objets mêlant la saisie de ces objets par Bret Victor.

FIGURE 3.12 : La relation de l'humain avec un outil ou un instrument par Bret Victor.

Lafon [Beaudouin-Lafon, 2000]. Un outil est fait pour augmenter les capacités de l'être humain, et non pas pour les réduire. Une de ces capacités qui précède un geste est la saisie d'objets.

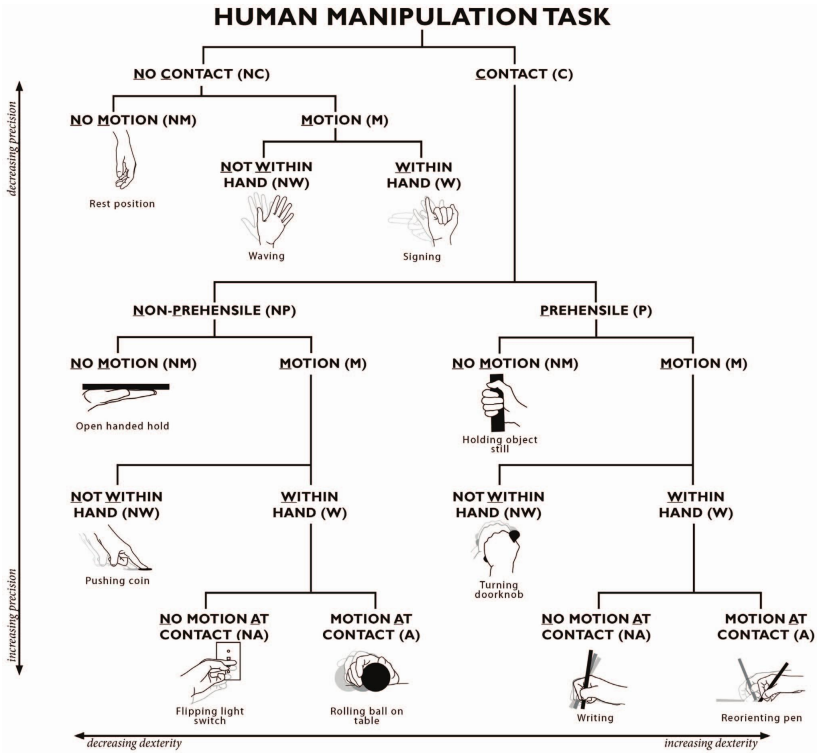


FIGURE 3.13 : Les gestes de manipulation de la main selon Bullock

Venant du domaine de la robotique, la taxonomie de Bullock et Dollar [Bullock and Dollar, 2011] permet de classifier les manipulations humaines en prenant comme élément central la main. La taxonomie des manipulations de Bullock dans la figure 3.13, utilise les mouvements de la main avec des sous-catégories de contact ou non avec l'objet, et pour les manipulations avec contact on trouve ceux avec ou sans la saisie d'objet. Dans la même classification, deux échelles sont utilisées pour placer les manipulations sur la carte. Elles quantifient d'une part la précision de la manipulation et d'autre part la dextérité (ou habileté).

La taxonomie de Bullock consiste à spécifier les manipulations humaines pour pouvoir s'inspirer des mêmes principes pour une réplication dans le

domaine de la robotique. Une mise à jour du travail une année après par le même auteur [Bullock et al., 2012] contient la même taxonomie mais avec des exemples de réplication dans des bras de robots pour la saisie des objets.

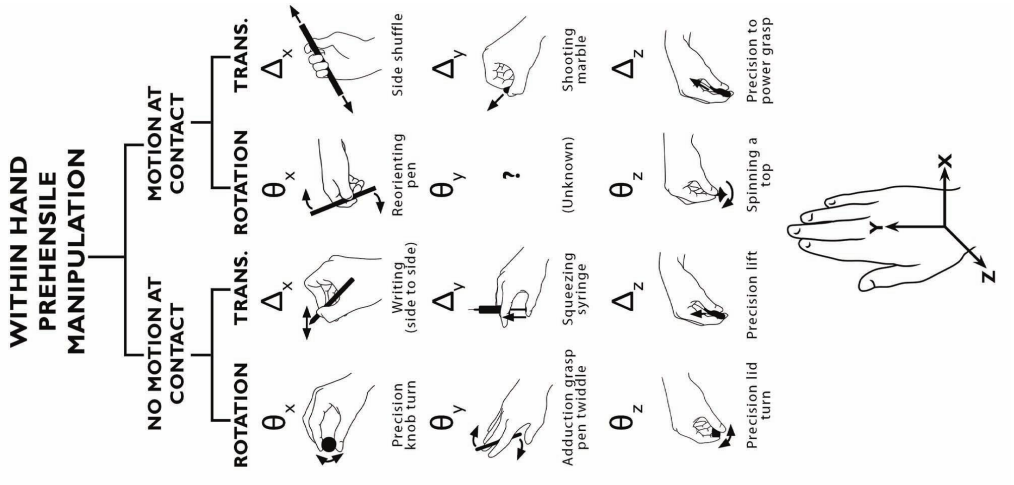


FIGURE 3.14 : La classification de Bullock des manipulations avec saisie d'objet selon l'axe de la main

Dans cette nouvelle classification des sous-groupes de manipulations des objets avec la main, les mouvements sont soit une rotation soit une translation tel que montre la figure3.14. Les mouvements de compression d'une seringue (suivant les différents axes), sont mis dans la translation parce que le point central de la classification est la main et pas forcément l'objet manipulé.

D'un point de vue qui se concentre sur la saisie pour des fins en robotique dans le travail de Mark Cutkosky [Cutkosky, 1989], l'auteur présente une critique des anciens modèles de saisie et de manipulation et leur utilisation limitée à des environnements d'industrialisation. Il présente une nouvelle taxonomie de la saisie humaine présenté par la figure **??** et y introduit la forme de l'objet, même si c'est d'une façon réduite à deux types : prismatique ou circulaire.

Le travail de Raphael Wimmer [Wimmer, 2011] étudie les aspects liés à la saisie et la façon de tenir un objet. L'auteur fait une étude sur les facteurs

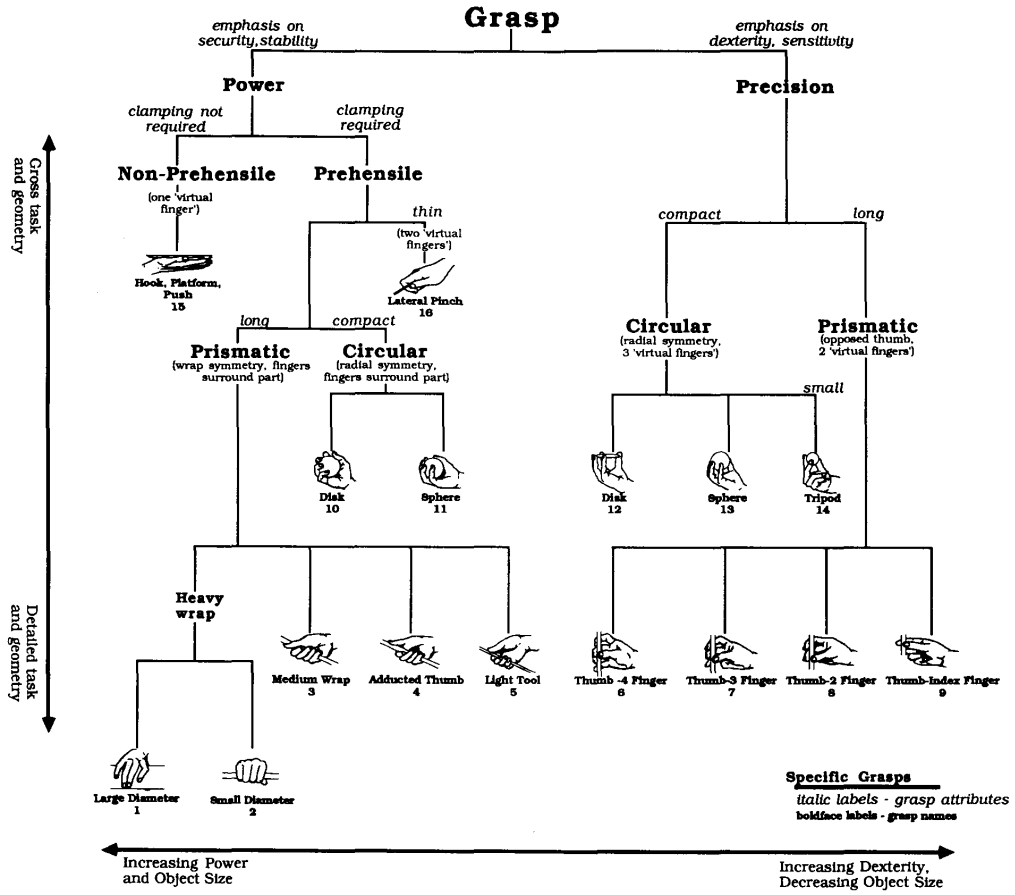


FIGURE 3.15 : La taxonomie de la saisie incluant la forme de l'objet reprise depuis [Cutkosky, 1989]

intervenants dans une opération de saisie avec des objets réels. Le travail relève des travaux menés en interaction tangible.

La saisie effectuée peut être utilisée pour collecter les informations sur le contexte, et peut aussi permettre l'amélioration de l'opération en créant le contexte spécifique pour un type de saisie voulue dans l'expérience. Le modèle de la saisie discuté gère les 5 facteurs : le but, la relation opérateur-objet, l'anatomie, les réglages et les propriétés de l'objet comme montre la figure 3.16.

Le livre de Christine MacKenzie et Thea Iberal [MacKenzie and Iberall, 1994] est remarquable dans le sujet de la saisie humaine. Dans ce livre on com-

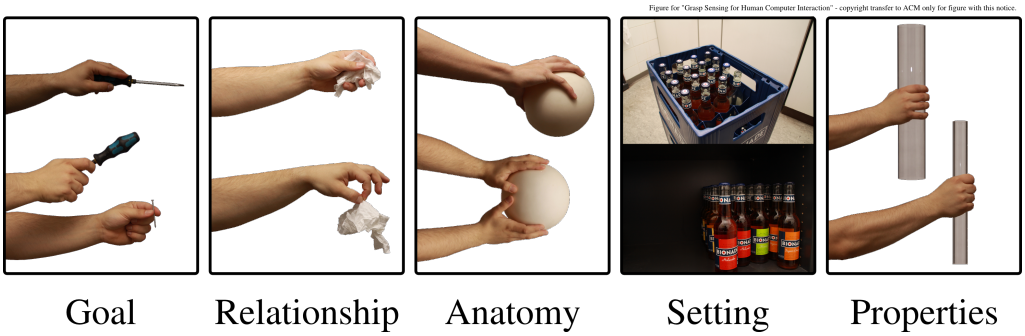


FIGURE 3.16 : Les facteurs influençant une saisie [Wimmer, 2011].

mence par la définition de la saisie des objets en considérant le but de les manipuler, les transporter voire même les sentir. Ceci semble bâtir les bases d'une opération d'interaction avec des objets qui commence évidemment avec la saisie de ce même objet.

3.8.1 Types de Saisie

Lors d'une saisie d'un objet, la main se déplace et prend un objet suivant une configuration liée à la forme géométrique et à la fonctionnalité de la saisie.

D'après les travaux de John Napier [Napier, 1956], les mouvements de la main autour d'un objet peuvent être divisés en 2 parties, les mouvements de préhension (prenante) et dans ce cas l'objet est saisi de façon partielle ou complète avec la main. Les mouvements de non préhension où la manipulation des objets n'intègre pas la saisie. Les objets peuvent être manipulés en les poussant ou en les levant par la totalité de la main ou les doigts.

John Napier classifie les saisies d'objets en 2 grandes familles, la saisie de force, et la saisie de précision. Dans la saisie de force, l'objet est maintenu entre l'opposition créée entre la paume de la main et les doigts. Le pouce applique aussi une opposition dans le plan de la paume. Dans la saisie de précision, l'objet est maintenu avec l'opposition entre les terminaison des doigts et le pouce. D'autres taxonomies existent comme celle de Feix et al [Feix et al., 2009] où il fait le tour des différentes taxonomies qui existent

dans le but de réduire la complexité mécanique des outils de saisie représentés. L'étude a pris en compte 33 types de saisie et les a classés de façon hiérarchique selon le type de saisie (de précision/de force) et l'inclusion du pouce.

3.8.2 Plans d'opposition

Lors d'une saisie, le positionnement des doigts de la main se fait principalement suivant deux plans permettant une saisie stable. Ces plans sont appelés les plans d'opposition. En prenant une forme quelconque, il est possible de prédire la position des doigts en cherchant d'abord les plans d'opposition possibles sur l'objet. Pour les formes géométriques simples et cubiques, les plans d'opposition sont faciles à prédire surtout en ayant connaissance de la position de la main. Les doigts virtuels sont une méthode associée à la notion des plans d'opposition pour guider les robots à saisir les objets. Les études sur les plans d'opposition et la saisie sont dans la majorité des cas liées au domaines de la robotique. Les chercheurs ont pour mission de construire des robots capables de reprendre la façon humaine de la saisie et savoir à partir des données des capteurs sélectionner les plans d'opposition pour avoir une saisie stable.

3.9 Conclusion

Dans ce chapitre nous avons recensé les différents composants humains qui ont un effet sur la génération du geste humain. Nous avons regroupé les différents travaux pour avoir une étude plus large et plus profonde des aspects gestuels de toute la chaîne au lieu d'une petite partie comme dans la figure 3.1. Nous sommes parti du cerveau pour comprendre les mécanismes internes de vision, mouvements, saisie et ensuite arrivant à la manipulation d'objets.

Deuxième partie

Technologies relatives au geste

Chapitre 4

Les technologies de reconnaissance du geste

4.1 Introduction

Chaque année, de nouveaux périphériques permettant la capture 2D, 3D ou la détection et la reconnaissance des gestes sont disponibles sur le marché. Chacun de ces dispositifs utilise une nouvelle méthode de capture spécifique ou recycle une ancienne technologie avec de nouveaux algorithmes améliorés. Dans ce chapitre, on recense les différents périphériques de capture utilisés dans le domaine gestuel. Notre sélection contient aussi les périphériques dont la fonction principale n'est pas la capture des gestes.

4.2 Les périphériques, les algorithmes et la main

Notre vision des périphériques autour de la reconnaissance des gestes est qu'ils permettent de traduire des propriétés physiques vers un type d'informations utilisable. Ils peuvent par exemple transmettre les informations sur la configuration de la main et ainsi extraire la posture et le geste. Chaque périphérique détecte à sa façon un sous-ensemble de paramètres selon la

technologie utilisée et ses capteurs. Les algorithmes contribuent soit directement comme en imagerie à la détection de la main et ses paramètres, soit à posteriori en les appliquant sur la trajectoire des doigts ou de leur centre de gravité.

La main est donc le centre d'intérêt de tous les périphériques et des algorithmes que nous allons présenter.

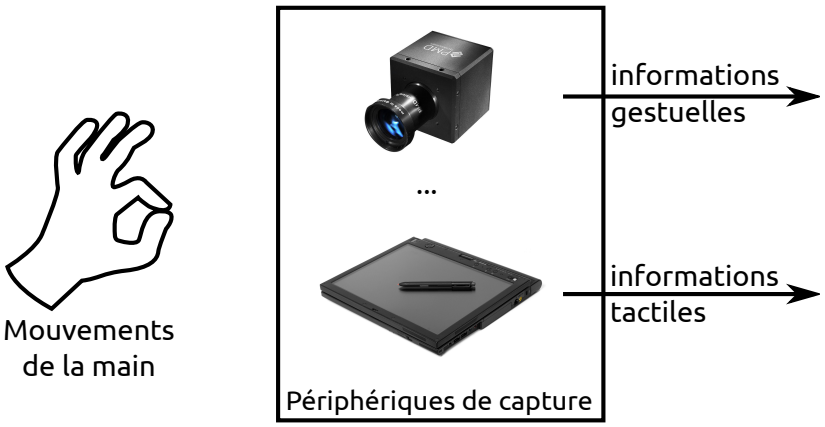


FIGURE 4.1 : Les périphériques jouant le rôle d'un filtre aux informations de la main

4.3 Les périphériques à détection mécanique ou à base de MEMS

4.3.1 Le Microphone

Il peut être étrange de noter que le microphone est un périphérique de reconnaissance des gestes. Mais avec quelques travaux récents, il est possible de l'utiliser pour détecter soit les fluctuations de l'air autour d'un périphérique, soit la façon avec laquelle on touche un écran quelconque. Puisqu'il n'arrive à capter que le son, il est nécessaire de l'associer avec des algorithmes puissants pour "voir" ce qui se cache dans un simple son.

Gupta et al. [Gupta et al., 2012] ont réussi à détecter cinq gestes basiques en utilisant l'effet Doppler généré par les mouvements de la main et transmis via l'air. Harrison et al. [Harrison et al., 2011] utilisent le microphone pour différencier entre les façons de cliquer sur un écran. Cette technique utilise la différence dans le son généré. L'utilisation du microphone permet d'élargir l'espace d'interaction gestuelle avec un coût assez faible et de nombreux autres travaux ont exploité cette flexibilité dont [Xiao et al., 2014, Harrison and Hudson, 2008].

4.3.2 La Wiimote

La Wii Remote ou la Wiimote est un périphérique jouant le rôle d'une manette de jeu. Elle contient principalement un capteur d'accélération pour détecter les mouvements, un capteur infrarouge pour détecter une orientation dans l'espace et la proximité de la télévision. La détection de la rotation et de la distance se fait en suivant au maximum quatre LED infrarouge à l'intérieur de la "Sensor Bar" ou équivalent. Avec des extensions comme le Wii MotionPlus, elle peut aussi avoir un gyroscope qui est utilisé pour améliorer le résultat de le suivi. Le gyroscope et l'accéléromètre sont généralement utilisés ensemble pour améliorer le résultat des trajectoires. [Kratz et al., 2013]

FIGURE 4.2 : La wiimote, le module MotionPlus, et le Nunchuck

4.3.3 Le GameTrak

Le GameTrak est un périphérique qui permet de suivre 2 points dans l'espace en utilisant un fil enroulé pour chaque cible. Ce fil se situe à la base et permet une fois déroulé par les mouvements de l'utilisateur de connaître la distance. D'autres capteurs permettent de connaître les angle à la base. La fusion des deux informations permet de savoir l'emplacement des points dans l'espace. Une version qui s'appelle GameTrak Freedom sans fil a été annoncée mais n'est jamais sortie. Cette nouvelle version utilise l'ultrason combiné avec les données d'un accéléromètre et utilise un processus appelé trilatération [1] pour le positionnement dans l'espace utilisé aussi par le système de positionnement GPS et similaire à la triangularisation mais sans utiliser les angles. Cette nouvelle version n'est jamais sortie. L'entreprise a été ensuite achetée par une autre appelée PDP.

4.3.4 Le Razer Hydra

La Razer Hydra est un périphérique qui permet de détecter la position et l'orientation de deux manettes dans l'espace. Elle utilise un système magnétique pour détecter la variation du champ et ainsi estimer la position et l'orientation 3D. Elle est utilisée surtout en réalité virtuelle pour se déplacer et interagir dans l'environnement 3D.

FIGURE 4.3 : La Razer Hydra

1. http ://fr.wikipedia.org/wiki/Trilatération

4.3.5 La Myo

La Myo développé par l'entreprise Thalmic Labs est un bracelet sans fil (bluetooth) qui détecte les impulsions électriques des muscles du bras appelées EMG ou électromyographie pour estimer les déplacements et les gestes de la main. À part les capteurs EMG, la Myo contient aussi d'autres capteurs dont un gyroscope, un accéléromètre et un magnétomètre. La fusion des informations des différents capteurs qui se fait dans le bracelet est la clé pour pouvoir détecter les gestes de l'utilisateur. La Myo contient un processeur ARM Cortex-M4 32 bits assez puissant et qui a des blocs de calcul spécifiques DSP [2].

FIGURE 4.4 : La Myo par Thalmic Labs

4.3.6 Génération des tracés

En utilisant ces périphériques, on arrive à détecter une trajectoire 3D dans l'espace. Cette trajectoire représente dans la majorité des cas la position du centre de gravité de la main en mouvement. Avoir ce tracé, est le minimum pour pouvoir ensuite appliquer de nouveaux algorithmes de détections des gestes. C'est l'étape de base pour commencer à parler des gestes.

2. Digital Signal Processor, ou processeur de traitement numérique du signal

4.4 Les périphériques à base de caméras

4.4.1 La Sony Playstation Eye

La camera USB de la Playstation-3 est encore l'un des périphériques les plus utilisés dans le domaine de la détection des gestes. La caméra ne coûte pas très cher, et malgré ceci, elle offre une résolution correcte de 640x480 pixels avec un résultat fidèle dans la transmission des couleurs surtout dans des conditions non idéales (peu de lumière). L'une des caractéristiques qui ont fait sa popularité est sa fréquence de 60Hz qui est le double de ce que les webcams peuvent faire dans les meilleures conditions. Cette même fréquence peut aller jusqu'à 120Hz en réduisant la résolution à 320x240. Elle est très utilisée dans les installations de tables tactiles après l'ajout d'un filtre infrarouge.

FIGURE 4.5 : La caméra de la playstation 3, PS3 Eye

La caméra de la dernière itération du PlayStation est maintenant deux caméras avec des performances améliorées pour générer la stéréoscopie. Par contre, la prise de branchement de son câble est propriétaire et ne pouvait plus être branchée directement sur un PC. Un utilisateur a réussi à la faire fonctionner sous Linux et MacOS en branchant manuellement les fils à un

Fréquence	PS3 Eye	PS4 Eye
60 Hz	640×480	1280×800
120 Hz	320×240	640×400
240 Hz	-	320×192

TABLE 4.1: Comparaison des performances des caméras de la Playstation 3 et 4

cable USB 3. Il a proposé un pilote [3] et des outils [4] pour acquérir les images des deux caméras. Une comparaison des performances des caméras des deux itérations de Playstation 3 et 4 est présentée par le tableau 4.1.

FIGURE 4.6 : La caméra stéréoscopique du Playstation 4

4.4.2 Les caméras modifiées

Les caméras normales utilisées pour transmettre une vidéo RGB peuvent être modifiées pour gérer les informations autrement, ou seulement la plage de fréquence nécessaire. Des travaux ont proposé l'ajout de LED infrarouges qui projettent sur les objets devant la caméra, le niveau de l'infrarouge retransmis permet de détecter et/ou focaliser sur les objets qui passent devant la caméra. Une caméra normale détecte un spectre incluant la lumière visible et la lumière infrarouge, on lui ajoute un filtre infrarouge pour se limiter aux objets éclairés par les LEDS. Le travail de Gandy et al.[Gandy et al., 2000] se positionne dans ce contexte. Sachant que pour la caméra PS3 Eye, celles

3. https ://github.com/psxdev/PS4EYECam
4. https ://github.com/ps4eye/ps4eye

avec une lentilles bombés comme dans la figure sont les plus faciles à être modifiées.

FIGURE 4.7 : Le capteur PS3Eye modifié et personnalisé en ajoutant un lentille télescopique

4.4.3 La Leapmotion

La leapmotion est un périphérique qui contient 3 LED et 2 caméras RGB. La résolution de chacune des caméras est de 640x240 en mode normal et 320x240 en mode robuste. La luminosité des LED varie selon le rapprochement de la main vers le capteur. Le module hardware ne fait que transmettre les images à des fréquences pouvant atteindre 300Hz vers l'ordinateur.

FIGURE 4.8 : Le capteur Leapmotion

Le traitement des données est fait en totalité sur l'ordinateur et prend parfois beaucoup de ressources sur une configuration à moyennes performances. L'algorithme utilisé crée une carte de disparité 3D à partir de deux

images illuminés par les LED infrarouge. L'algorithme utilise des heuristiques pour accélérer la génération de cette carte qui ne fonctionne que pour la main. L'algorithme interne de la Leapmotion a été partiellement exposé via une mise-à-jour non intentionnée. Avec un test sur un autre objet 3D ou avec le visage, l'algorithme étend le nez sur toute la zone de détection. La création du modèle 3D de la main se prépare avec une version adapté de l'algorithme k-means qui fait correspondre des sphères dans chaque doigt.

4.4.4 Le module de vision PiCam

L'entreprise Pelican Imaging a lancé une nouvelle technologie permettant de générer une vision d'image 3D. Cette technologie se base sur une matrice carrée de taille 4 de micros caméras. Cette matrice décrite dans l'article de Venkataraman et al. [Venkataraman et al., 2013] construit la 3D grâce à un algorithme d'extrapolation des différences entre chacune des caméras qui sont d'une faible résolution. Vu que cette technologie n'est pas encore en vente grand public au moment de la rédaction de ce manuscrit, elle n'a pas été utilisée pour la reconnaissance des gestes. Par contre, elle peut prendre une part du marché vu la faible dimensions de ces capteurs.

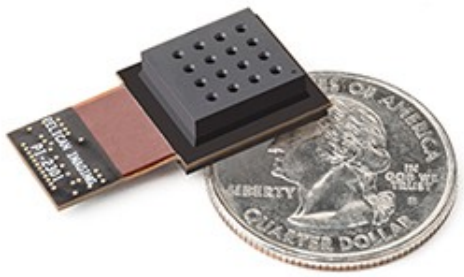


FIGURE 4.9 : Le module de capture PiCam de Pelican Imaging

4.4.5 La Kinect 1, projection en lumière structurée

La première version de la Kinect est un périphérique qui possède une camera RGB et un système de projection 3D en lumière structurée. Le système émet un grille de 9 carreaux, chaque carreau contient un motif spécifique de points en infrarouge comme dans la figure 4.10 extraite avec un appareil photo et un filtre infrarouge [5]. La projection des points sur la surface et la vision binoculaire permettent de générer "la carte de disparité" et de détecter la profondeur.

La Kinect contient aussi un moteur pour régler l'angle des caméras et un capteur d'orientation. Ces additions nécessitent une alimentation extérieure contrairement à un périphérique similaire appelé Asus Xtion et qui ne contient que le module de la détection de la 3D de la Kinect et qui se branche directement sur la prise USB 2.

4.4.6 Les caméras à temps de vol

4.4.6.1 La Kinect 2

La deuxième version de la Kinect arrive avec une nouvelle technologie de reconnaissance. Elle rejoint la famille des caméras Time-of-Flight ou TOF. Il n'y a plus de moteur ou de capteur d'orientation mais une camera RGB en résolution Full-HD (1080p) et un émetteur de lumière. La profondeur est détectée par le temps que prend la lumière pour faire l'aller-retour vers la cible. La Kinect 2 nécessite une alimentation et un port USB3 pour pouvoir fonctionner. La résolution du capteur 3D de la Kinect 2 est de 512x424 points.

4.4.6.2 La caméra Intel RealSense

La camera RealSense par Intel est aussi une caméra Time-of-Flight créé suite à une collaboration entre Intel et Creative Technology. Elle utilise un

5. http ://www.futurepicture.org/ ?p=97

FIGURE 4.10 : La première version de la Kinect, et la visualisation du motif infrarouge émis et utilisé pour la capture 3D

FIGURE 4.11 : La deuxième version de la Kinect (temps-de-vol)

module créé par SoftKinetic parvenant de la DepthSense 325. La caméra peut être comparée à une Kinect2 mais en version miniature.

FIGURE 4.12 : Les caméras Intel RealSense

4.4.6.3 La caméra Cambord Pico

PMD Technologies est une entreprise qui travaille dans le domaine de la capture 3D depuis 1996 sous le nom de S-Tech, puis en 2002 avec son nom actuel. Elle construit des caméras à temps de vol pour différentes applications qui n'étaient pas liées au domaine de l'IHM. Au début, les applications étaient industrielles pures pour la capture du niveau d'huile dans un réservoir. Actuellement, PMD Technologies vend une caméra d'une résolution de 160x120 pixels.

4.4.7 Les caméras infrarouges professionnelles

Ce genre de caméras se distingue par sa précision mais aussi son coût élevé. La capture ne se fait pas en utilisant une seule caméra mais en utilisant plusieurs en même temps qui sont calibrées au début. Les caméras, entourés par des LED infrarouges, capturent des petites boules réfléchissantes attachées à des objets et éclairés par ces LED ou à une veste portée par un acteur. Le système contenant plusieurs caméras permet de connaître les positions et orientations en 3D. Ce genre de systèmes comme celui construit par les entreprises Vicon, Optitrack, ou ARTTRACK sont utilisés dans les studios de captures professionnels de cinémas.

4.4.8 Les caméras de détection de chaleur

Le prix des caméras qui permettent de détecter la chaleur sont actuellement en baisse. Leurs utilisation est de plus en plus fréquente que ce soit seule, ou avec la présence des caméras RGB ou de profondeur. Dans beaucoup de cas, ils permettent de détecter les utilisateurs sans avoir à faire beaucoup de traitements mais juste en faisant une segmentation sur la chaleur de la peau humaine. Dans les travail de Kurz [Kurz, 2014] ou de Larson [Larson et al., 2011], elle ont été utilisées pour détecter les gestes à travers la chaleur laissée par les doigts sur une surface. Un kit de développement contenant un capteur FLIR [6] appelé "Lepton longwave infrared (LWIR) imager" est en vente à moins de 350 euros. La caméra a une faible résolution de 80x60 pixels mais elle a une sensitivité de moins de 0.05 Kelvin [7]. Dans le travail de [Zeng et al., 2012], une caméra a été utilisée pour faciliter la suppression de l'arrière plan pour ne laisser que les zones intéressantes sans utiliser des algorithmes complexes et coûteux en terme du temps d'exécution avec seulement la caméra RGB. Dans le travail de [Saba et al., 2012], la caméra de chaleur a été utilisé conjointement avec la caméra de profondeur Kinect pour détecter le contact avec la surface d'une table et informer sur le niveau de pression utilisé.

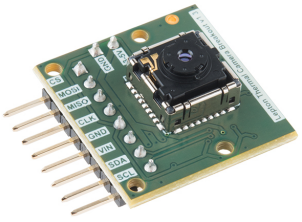


FIGURE 4.13 : Le module de capture FLIR

6. Forward Looking Infrared
7. https ://www.sparkfun.com/products/13233

4.5 Synthèse des périphériques de capture

Chaque gamme de périphériques décrite précédemment a une méthode de capture différente des autres. Dans cette synthèse nous les classifions en proposant des critères de comparaison partagés. L'évolution des capteurs dans les dernières années a pris un rythme exponentiel. Que ce soit en terme de réduction de prix de vente ou sur les caractéristiques techniques.

4.5.1 Évolution des performances et de la précision

Les caméras 3D basées sur la technologie temps de vol ont eu une amélioration visible depuis une dizaine d'année. La figure suivante montre un aspect de l'évolution. La technologie a commencé avec un capteur sur une seule dimension, permettant par exemple de savoir le niveau d'un liquide dans un réservoir et qui peut être vu comme une résolution d'un seul pixel. C'était le cas avec l'entreprise PMD Technologies déjà cité pendant 1998, jusqu'à arriver en 2014 à une résolution de 512x424 points sur la Kinect version 2.

-	Kinect1	Kinect2	Realsense	Camboard	Leapmotion	PS Eye
Fréq.(Hz)	30	30	60	90	90	60

TABLE 4.2: Comparaison des fréquences des périphériques

La fréquence de transmission des informations varie selon le capteur utilisé. Si on a des contraintes temps-réel humaines rigides, il est important de choisir un périphérique avec une grande fréquence de transmission. Le tableau 4.2 montre les fréquences des dispositifs à base de caméras. Il est important de noter aussi que si la sortie est une structure de bas niveau, il faut compter le temps de traitement nécessaire à les adapter pour être utilisée dans le contexte d'une reconnaissance de gestes.

4.5.2 Zones de capture

La zone de capture d'un périphérique est le facteur clé dans un choix du périphérique à utiliser. La zone de capture n'est pas généralement rectangulaire. Pour les périphériques de détection 3D, si la distance est trop proche ou trop éloignée, aucune information n'est transmise. La figure 4.14 donne un aperçu des zones à partir desquels le périphérique commence à transmettre les informations, et ses limites de distance. On voit clairement que la Kinect dans toutes ces versions permet de capturer une zone beaucoup plus large que les autres capteurs.

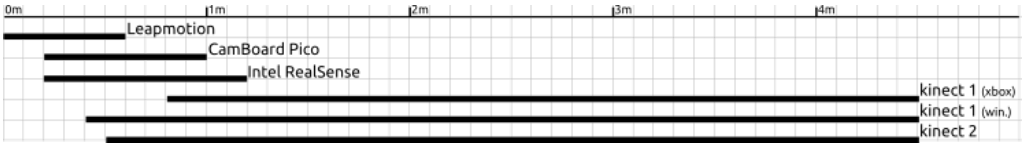


FIGURE 4.14 : Les zones de captures de chacun des périphériques offrant une sortie 3D

Chacune des caméras a une ouverture horizontale et verticale exprimée en angle de champ de vision en horizontal et en vertical. Ces ouvertures définissent aussi les limites du trapézoïde 3D de capture.

4.5.3 Coûts

Les coûts des capteurs sont aussi un facteur dans le choix d'utiliser un périphérique ou un autre.

Les caméras RGB avec une bonne résolution et une bonne fréquence sont chers, la PS3 Eye est la moins cher qu'on peut trouver qui peut aller à des fréquence de 60Hz ou plus. La PS4 Eye elle par contre coûte dans les 50 euros pour une résolution plus grande et une fréquence pouvant aller jusqu'à 240 Hz. Cette dernière n'est pas utilisable directement, mais nécessite une opération de recâblage pour transformer la prise propriétaire vers une prise USB3. Une fois transformé, c'est le capteur RGB le moins cher sur le marché qui fournit des performances haut de gamme à un prix minimal.

Les technologies de capteurs à temps de vols sont de moins en moins coûteux depuis ces dernières années. Les premières versions de la caméra Camboard Pico étaient facturés à plus de 500 euros. Grâce à la production en grande échelle, elle coûte dans les 100 euros en 2015. La Kinect 2 avec sa résolution de 512x424 points est aussi la caméra temps-de-vol la moins chère dans le marché en 2015. Les autres caméras professionnelles ont une résolution inférieure ou sont beaucoup plus chers.

Les caméras de détection de chaleur ou FLIR ont commencé elles aussi la chute des prix avec le module à moins de 350 dollars en 2015 pour la Leption présenté dans le paragraphe 4.4.8. Les anciennes applications étant en grande partie liées au domaine militaire ou de recherche et n'avaient pas un prix pour une utilisation en grand public.

4.6 Conclusion

Dans ce chapitre, on a fait le recensement des différents périphériques de capture en les classifiant dans des catégories. La synthèse nous a permis de choisir lesquelles utiliser en fonction de l'accès à des données bas niveau et ainsi pouvoir appliquer des algorithmes additionnels sans être attachés à ce que le périphérique nous fournit en haut niveau. Les différences sur la zone de capture possible et la perturbation par rapport à l'environnement d'utilisation éliminent certains choix qui semblent être fonctionnels à première vue. Notre choix s'est porté sur l'utilisation de la première version de la Kinect vu sa couverture, accès aux données bas niveau, et sa disponibilité en 2011 date de début de la thèse. En 2014 le choix aurait dû se porter sur l'utilisation de la Kinect 2.

Chapitre 5

Algorithmes de reconnaissance

5.1 Introduction

Dans le chapitre précédent, nous avons fait le tour des différents périphériques disponibles et qui peuvent être utilisés dans l'opération de capture des gestes 2D ou 3D. Même si certains périphériques transmettent des résultats pré-traités et directement utilisables comme les informations des mains reçus à travers la Leapmotion, certains autres nécessitent plusieurs blocs algorithmiques de traitement pour arriver à un résultat utilisable. Dans ce chapitre on va présenter une sélection des algorithmes les plus utilisés dans les pré-traitements ou la reconnaissance des gestes et qui peuvent être intégrés déjà dans un système similaire à la Leapmotion.

L'article d'état de l'art de Pavlovic et al [Pavlovic et al., 1997] présente une large partie des algorithmes, les modèles et les manières de gérer la reconnaissance avec ou sans la modélisation de la main. Il différencie les systèmes qui construisent un modèle 3D de la main, et les systèmes qui utilisent un modèle d'apparence. Dans le premier cas, les résultats sont meilleurs mais il n'est pas possible de les utiliser avec des contraintes temps-réel, et dans le deuxième cas, on n'a pas la maniabilité générale et les solutions fonctionnent pour des cas spécifiques.

5.2 Algorithmes de reconnaissance des tracés

Les tracés qui représentent un mouvement dans l'espace 2D ou 3D sont fournis par les périphériques de captures. C'est un moyen répandu pour reconnaître les gestes. Les algorithmes les plus matures et les plus utilisés sont ceux conçus pour la reconnaissance des tracés de traits simples sous formes d'une série de points dans l'espace ou sur le plan. Ces tracés sont bidimensionnels et générés avec un stylet ou sur une table tactile. Les études sur les tracés 3D sont très rares.

5.2.1 Les algorithmes en Soft Computing

Il existe un grand ensemble d'algorithmes qui peuvent être appliqués pour reconnaître les gestes ou même un comportement générique inspirés par les méthodes humaines ou naturelles pour résoudre un problème. Cet ensemble d'algorithmes est appelé "algorithmes de Soft Computing". Les chaînes de Markov cachées, les systèmes à logique floue, les réseaux de neurones et les algorithmes évolutionnistes font partie de cet ensemble. Une partie des algorithmes de ce genre a été présenté dans l'article de synthèse de Mitra [Mitra and Acharya, 2007] citant les chaînes de Markov cachées les réseaux de neurones multi-couches

5.2.2 Les algorithmes géométriques

La thèse de Dean Rubine [Rubine, 1991] fut l'une des premières à traiter le sujet en 1991. Elle utilise à travers une approche statistique les caractéristiques géométriques d'un tracé comme montre la figure 5.1 pour les utiliser dans la reconnaissance. Les travaux décrits présentent la reconnaissance des gestes composés d'un seul tracé, ensuite la prédiction du geste avant qu'il soit finit en exécutant l'algorithme plusieurs fois avant la fin du

traçage et tant que le résultat est encore ambiguë. Rubine a aussi traité la reconnaissance des tracés composés de plusieurs chemins.

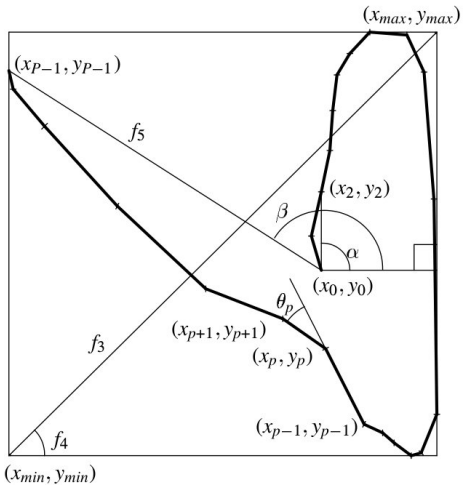


FIGURE 5.1 : Extraction des caractéristiques géométriques d'un tracé gestuelle selon la thèse de Rubine

Un travail remarquable, basé sur les algorithmes de Rubine et sur la reconnaissance prématuré des tracés, est Octopocus [Bau and Mackay, 2008]. Dans ce travail, les chercheurs ont construit un outil de visualisation de la reconnaissance de gestes qui ne nécessite pas d'apprendre par cœur les gestes possibles. Selon le principe du feedforward, l'utilisateur de ce système a un retour visuel des tracés de gestes qui partagent le même début du tracé initial que le tracé en cours de réalisation comme montre la figure 5.2.

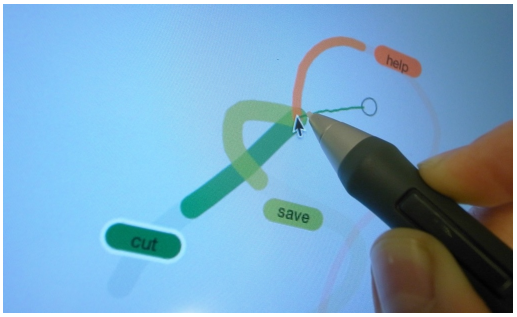


FIGURE 5.2 : Visualisation des gestes possibles avant la fin du geste actuel

Il existe des algorithmes de reconnaissance faisant intervenir les notions de réseaux de neurones ou les chaînes cachés de Markov qui sont des notions nécessitant un apprentissage, mais il existe aussi des algorithmes simples et efficaces pour reconnaître les tracés comme l'algorithme $1 réalisé par Wobbrock et al. [Wobbrock et al., 2007]. L'algorithme se base sur des modèles pré-enregistrés après normalisation. Les étapes de normalisation consistent d'abord à ré-échantillonner les points des tracés pour supprimer les effets de vitesse. Tous les tracés ont le même nombre de points, généralement 64 ou 128 points comme montre la figure 5.3.

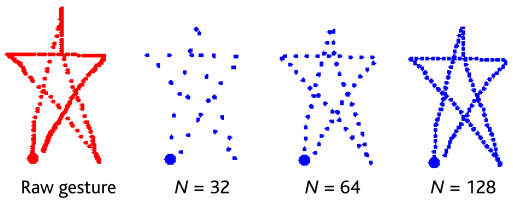


FIGURE 5.3 : Première étape de normalisation de l'algorithme 1$

Après cette étape, l'algorithme normalise les gestes vers un angle standard en calculant le centre de gravité du tracé puis en supprimant l'angle fait par l'horizontale et la ligne passant entre le centre et le premier point du geste comme montre la figure 5.3. Dans la dernière étape de normalisation, on change la forme du tracé pour obtenir une taille carrée et on place son centre sur l'origine du repère.

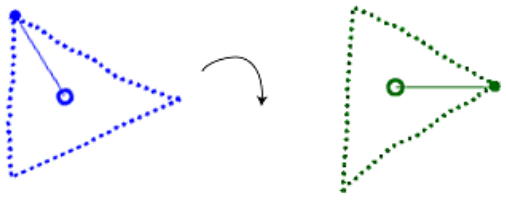


FIGURE 5.4 : La rotation du geste vers un angle initial de l'algorithme 1$

Finalement, on compare le geste entré par l'utilisateur à tous les modèles. La comparaison se fait en calculant la somme des distances euclidiennes qui séparent les points du modèle à celui du geste alignés temporel-

lement. Cette somme d'erreurs sera divisée par le nombre total de points. Plus le geste est différent de celui rentré, plus l'erreur est grande.

Un indice de correspondance est définit par $1 - erreur$. Ainsi, le tracé le plus proche de 1 sera pris en tant que geste reconnu. Malgré la simplicité de l'algorithme, il s'avère qu'il a des taux de réussite supérieurs à 90%.

L'algorithme a un certain succès en utilisation mais la dernière étape de comparaison en calculant la distance euclidienne aux multiples points constituants le tracé prend un temps de calcul qui doit être optimisé, surtout pour les appareils mobiles. Une évolution de l'algorithme appelée Protractor [Li, 2010] utilise la similarité ou la mesure de distance cosinus. La distance entre les deux tracés est donc l'angle entre les deux vecteurs représentants les tracés, calculé dans la n-ième dimension. Ceci a permis d'optimiser une partie du calcul.

$$S(t,g) = \frac{1}{\arccos \frac{v_t.v_g}{|v_t|.|v_g|}} \tag{5.1}$$

avec

$$v_t.v_g = \sum_{i=1}^{n}(x_{ti}x_{gi} + y_{ti}y_{gi}) \tag{5.2}$$

$$|v_t|.|v_g| = \sqrt{\sum_{i=1}^{n}(x_{ti}^2 + y_{ti}^2)}\sqrt{\sum_{i=1}^{n}(x_{gi}^2 + y_{gi}^2)} \tag{5.3}$$

L'algorithme $3 de Sven Kratz et Michael Rohs [Kratz and Rohs, 2010] utilise également des opérations géométriques pour détecter des gestes à partir de modèles pré-enregistrés. Le travail est basé sur les travaux de Wobbrock et al. [Wobbrock et al., 2007] et l'apport est dans l'application de l'algorithme sur les périphériques utilisant des accéléromètres. Puisque les informations d'accélérations se définissent en utilisant 3 axes, la normalisation s'effectue dans un cube au lieu d'un carré. Ils ont ensuite étendus leurs travaux pour créer Protractor3D qui ajoute l'utilisation des quaternions dans l'étape de normalisation pour la rotation.

Vu les performances et la simplicité de l'algorithme 1$, il a été repris comme une base de réflexion pour d'autres algorithmes le généralisant pour pouvoir reconnaître plusieurs tracés [Anthony and Wobbrock, 2010] ou en modifiant la façon avec laquelle il compare deux tracés en faisant la correspondance entre deux nuages de points [Vatavu et al., 2012]. Ce dernier travail montré par la figure 5.5 élimine la nécessité de savoir l'ordre de la production du ou des tracés en considérant son entrée comme un nuage de points qu'il faut faire correspondre avec un nuage de référence dans sa base de modèles.

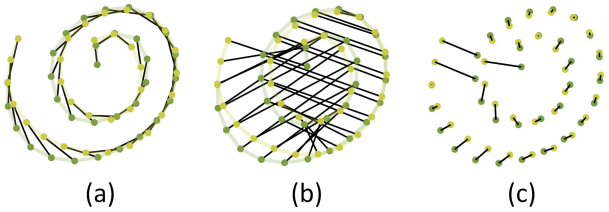
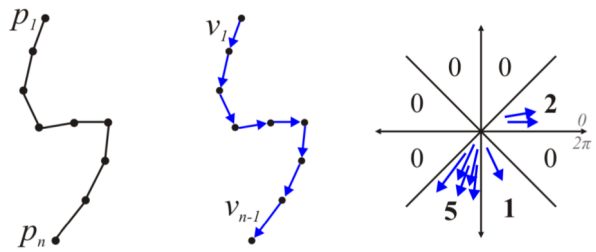


FIGURE 5.5 : La correspondance dans le calcul des distances pour l'algorithme $1 dans (a), $N dans (b), et $P dans (c)

L'algorithme de quantification des angles cité dans l'article d'Olsen et al. [Olsen et al., 2007] permet de construire rapidement un vecteur représentant le geste entré. Son principe est qu'après une phase de normalisation, on transforme les points en une série de vecteurs. On classifie ces vecteurs selon l'angle que fait chacun dans le cercle trigonométrique divisé sur 8 zones de 45°.

On construit un tableau contenant 8 valeurs représentant le nombre des vecteurs qu'on peut placer dans chaque zone. On divise à la fin par le nombre total de vecteurs pour avoir la somme des valeurs des éléments du tableau égale à 1. Toutes ces étapes sont montrés dans la figure 5.6 pour préparer les tableaux d'indices.

Cet algorithme est facile à comprendre et implémenter. Il permet aussi de générer des signatures qui peuvent être invariantes à la rotation. Par contre, il a le défaut de ne pas distinguer entre un tracé, et une répétition



(a) (b) (c)

(d) $\langle 2, 0, 0, 0, 0, 5, 1, 0 \rangle$ (e) $\langle \frac{1}{4}, 0, 0, 0, 0, \frac{5}{8}, \frac{1}{8}, 0 \rangle$

FIGURE 5.6 : Les étapes de traitement d'un tracé pour créer un tableau d'indices normalisés

de plusieurs fois de ce même tracé. Donc dans sa version de base, il génère la même signature comme le montre la figure 5.7.

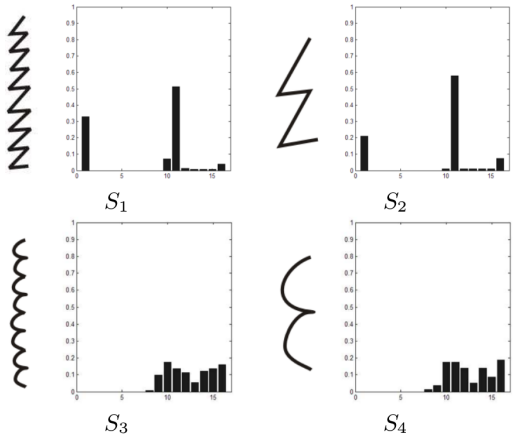


FIGURE 5.7 : L'algorithme de quantification des angles ne permet pas de faire la distinction entre un motif, et sa répétition

5.3 Geste multitactile

La démocratisation de l'utilisation des périphériques multi-tactiles a poussé les laboratoires de recherches ainsi que les entreprises à créer des systèmes pour mieux reconnaître les gestes de l'utilisateur. Les premiers

algorithmes reposent sur des notions reprises aux algorithmes de reconnaissance des gestes avec un seul trait, mais les contraintes d'utilisation grand public ont poussé loin les demandes et les spécifications requises.

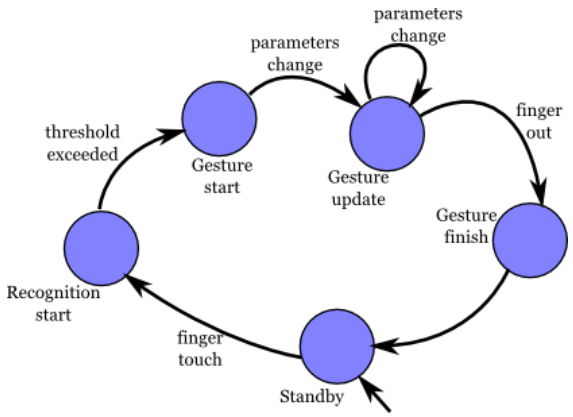


FIGURE 5.8 : Machine à états de la reconnaissance des gestes

Les algorithmes devaient reconnaître non plus des gestes mais des manipulations. Ce qui signifie être réactif aux doigts des utilisateurs et pouvoir décider rapidement de la nature du geste qui a été entré. Le moteur de reconnaissance développé dans le système d'exploitation ubuntu est l'un des moteurs les plus opérationnels tout en étant disponible librement en open source et dont j'ai fait partie de son développement. La figure 5.8 montre la machine à états finis utilisée pour déclencher la reconnaissance des gestes, qui commence avec le contact des doigts sur le support et finit avec leur levée. Les gestes reconnus sont en fait des micros gestes déclenchés après que le déplacement d'un ou de plusieurs doigts dépasse un certain seuil.

Ce moteur permet de reconnaître seulement 4 familles de gestes principaux qui sont le toucher, le déplacement, le zoom et la rotation. Chaque famille contient des sous-paramètres qui permettent d'extraire les autres types de gestes. Ces familles représentées par la figure 5.9 peuvent être déclenchées en parallèle, et c'est le rôle de l'application qui utilise le système de s'inscrire seulement à ceux dont elle a besoin.

Le moteur décrit permet de reconnaître les micro-gestes effectués pour

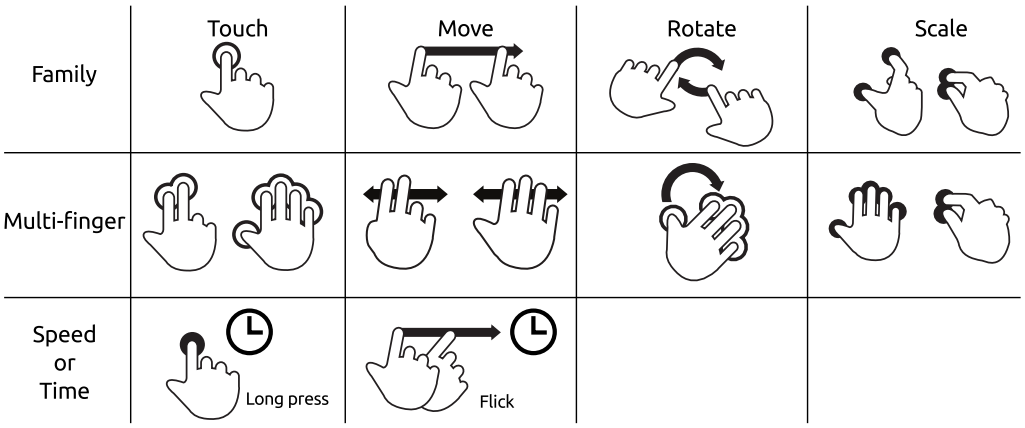


FIGURE 5.9 : Les familles de gestes multi-tactiles reconnus

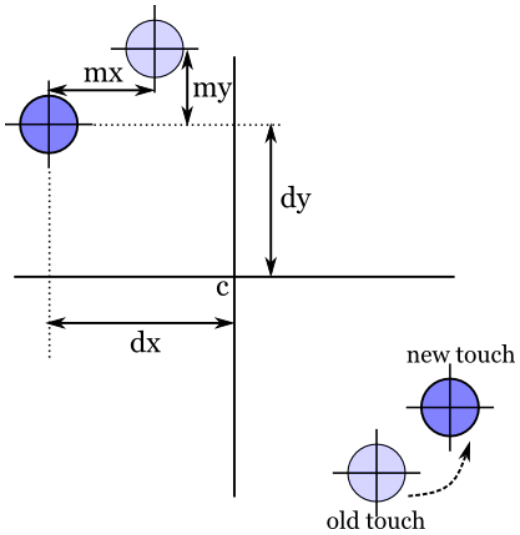


FIGURE 5.10 : Calcul de la rotation dans le moteur de reconnaissance des gestes tactiles

un nombre de doigts qui peut aller jusqu'à 10. La détection de la rotation est l'un des sujets qui peuvent être difficile à gérer pour une telle généricité. Le moteur arrive à détecter le geste de rotation en calculant la somme des delta des micro-rotation de chaque doigts comme le montre la figure 5.10. La valeur "rotation" est exprimée avec les équations suivantes.

$$rotation = \frac{1}{r^2}.\frac{1}{n}\sum^{n}(dx_i.my_i - dy_i.mx_i) \tag{5.4}$$

avec

$$r = \sqrt{\frac{1}{n}\sum^{n}(dx_i^2 - dy_i^2)} \tag{5.5}$$

Il existe d'autres approches pour la reconnaissance gestuelle multi-tactiles comme le travail de Kin et al. [Kin et al., 2012] où il traite les gestes multi-tactiles comme des expressions régulières. Cette approche permet de simplifier la création et la reconnaissance des gestes complexes. La figure 5.11 tirée de l'article cité, montre le principe.

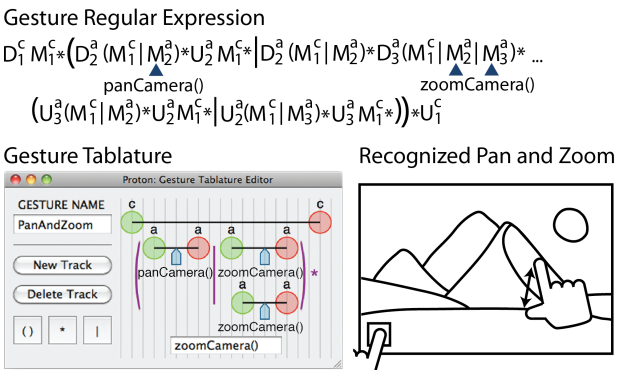


FIGURE 5.11 : Le moteur de création et de détection des gestes en les considérant comme des expression régulières Proton

5.4 Les algorithmes de reconnaissance en imagerie

En recevant un flux brut de la caméra, il est nécessaire de le traiter pour extraire les informations utiles pouvant aider à la reconnaissance d'un geste. Il y a une multitude d'approches pour l'extraction des informations qui peuvent parfois être invisibles pour l'humain, mais qui existent dans la scène.

5.4.1 La détection du flux optique

L'algorithme de détection du flux optique est l'un des plus simples qu'on puisse trouver pour détecter un objet qui bouge devant la caméra. Son principe est simple, c'est de détecter les vecteurs des mouvements par rapport au changement de pixels d'une image à un instant avec la précédente. L'algorithme de base se trouve implémenté dans la bibliothèque OpenCV et utilise la méthode de Lucas-Kanade [Lucas, 1985].

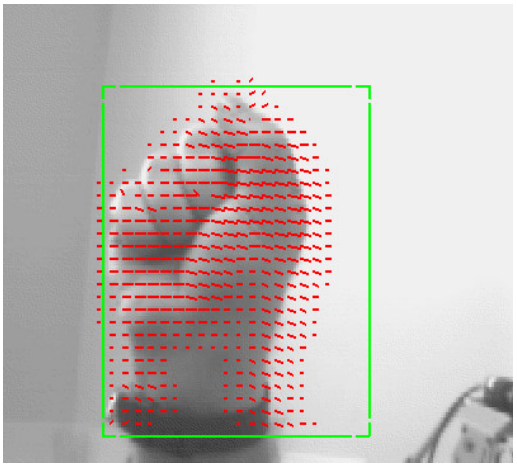


FIGURE 5.12 : Photo montrant l'utilisation du flux optique pour suivre la main [1]

1. http ://www.cim.mcgill.ca/ apl/Research/OpticalFlow/benoits.html

Dans son utilisation dans la détection du geste avec la main, son point faible est la perte de l'objet suivi après un certain temps. Cet algorithme peut être combiné à des variations ne suivant que le flux d'une partie de la vidéo ou qui réinitialise les zones de détection.

5.4.2 Détection de la peau humaine

En choisissant d'utiliser une caméra RGB normale tout en cherchant à détecter les gestes, la première étape à envisager est de détecter la main, et ceci en focalisant sur la couleur de la peau humaine. La procédure s'appelle segmentation, et se fait en éliminant les couleurs qui ne font pas partie de la zone de la peau humaine. La segmentation peut être précédée par le changement de l'espace de couleurs du RGB vers un autre plus simple à utiliser pour la segmentation comme le YCrCb ou le HSV [Boulabiar et al., 2011].

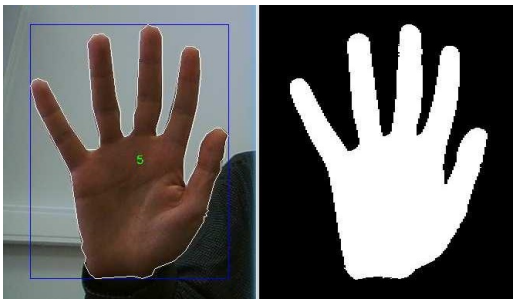


FIGURE 5.13 : Extraction de la peau humaine depuis une caméra et génération du contour

La détection de la zone avec la peau humaine conduira à avoir des zones binaires qui représentent la main, ensuite il faut utiliser d'autres algorithmes pour extraire plus d'informations comme le contour.

5.4.3 Utilisation des gants et des patchs colorés

Les algorithmes de détection de la peau humaine ont des limites d'efficacité surtout si la couleur de la peau devant la caméra est d'une teinte trop claire ou trop sombre par rapport à la section choisie dans l'espace

de couleurs. L'utilisation des gants colorés permet de faciliter l'étape de segmentation puisque on a déjà des couleurs fixes et prédéfinies. Le projet "SixthSense" de Mistry et al. [Mistry et al., 2009] utilise 4 patch colorés à mettre sur les doigts. Ils sont pris pour détecter certaines positions et activer des actions de capture ou autre selon le cas.

Le travail de Robert Wang [Wang and Popović, 2009] utilise des gants avec plusieurs patchs colorés disposés et calibrés à priori. Sa méthode comme le montre la figure 5.14 consiste à enregistrer toutes les formes possibles de mains dans une base de données et les récupérer par comparaison à la forme qui est capté en temps-réel. Pour comparer les posture enregistrés avec celle qui sont capturés, il a d'abord utilisé une distance similaire à celle de Hausdorff [2].

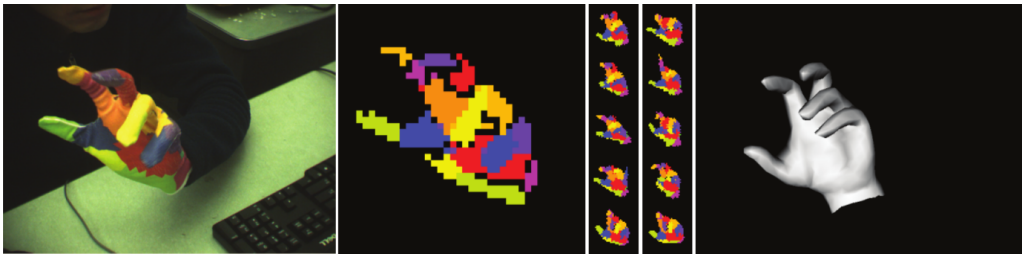


FIGURE 5.14 : Utilisant des gants colorés par Wang pour la détection de la posture de la main

Pour faire rapidement, il enregistre une capture comme une petite image, transforme l'image en un code binaire de 128 bits, ensuite utilise une mesure de distance de Hamming [3]pour chercher le code le plus proche à celui détecté et ainsi récupérer la forme de la main qui en découle.

5.4.4 Amplification des critères invisibles

Une vidéo capturée par la caméra peut contenir plus d'informations que ce que l'œil humain peut détecter. Les petites vibrations peuvent exister dans la vidéo à travers des changements minuscules. Le travail de Wu

2. http ://fr.wikipedia.org/wiki/Distance_de_Hausdorff
3. http ://fr.wikipedia.org/wiki/Distance_de_Hamming

et al. [Wu et al., 2012] du MIT permet d'amplifier les petits changements comme les effets du battement du cœur sur le gonflement d'une veine de la main, ou le changement des couleurs du visage, ou la respiration avec le gonflement du torse. Ces petits changements peuvent être utilisés pour détecter le changement de la couleur du haut de l'extrémité du doigt lorsqu'on touche une surface. Cette technique a été ensuite amélioré par Wadhwa et al. [Wadhwa et al., 2013] en utilisant les valeurs complexes de la représentation multi-résolution de l'image ou représentation pyramidale de l'image. La nouvelle technique est plus résistance au bruit. Ces deux algorithmes pour l'amplification des critères invisibles peuvent être utilisés pour amplifier la couleur du doigt en train de toucher une surface pour détecter la force de l'appui ou détecter s'il y a eu un appui tout court.

5.4.5 Recalage d'un nuage de points

Pour détecter une configuration de la main en 3D, une des méthodes est d'essayer de recaler un modèle de la main sur le nuage de points capturé en temps réel. Un modèle de la main est nécessaire pour pouvoir faire le recalage puisque la forme de la main change et il faut bien savoir à travers le modèle comment positionner chaque zone du modèle sur l'entrée brute qui est le nuage de points.

On notera le travail de Oikonomidis et al. pour générer le calibrer le modèle d'une seule main [Oikonomidis et al., 2011a] ou de deux [Oikonomidis et al., 2012] et avec la présence d'objets dans la scène [Oikonomidis et al., 2011b]. Le problème est traité comme celui d'une optimisation utilisant une version modifiée de l'algorithme d'optimisation par essaims particulaires [4] et exécuté sur la carte graphique. Vu que c'est une méthode d'optimisation et recalage avec un modèle, elle est encore lente à s'exécuter même en optimisant l'algorithme pour pouvoir s'exécuter sur une carte graphique, sa fréquence actuelle est de 15 Hz et nécessite donc une configuration spéciale.

4. http ://fr.wikipedia.org/wiki/Optimisation_par_essaims_particulaires

5.4.6 Critères de comparaison

Pour détecter un geste ou une posture, il y a un minimum de critères et d'informations à extraire pour pouvoir comparer à un modèle. L'un de ces critères est la détection des mouvements de la main au cours du temps, mais aussi la détection de son contour. Dans le cas d'une entrée avec une caméra RGB, après avoir séparé la main du reste de l'image, il faut d'abord suivre le blob tout au cours du temps. Un blob est une zone de pixels qu'il est intéressant de suivre. C'est le résultat qui reste de l'étape de segmentation.

Le travail de Michel et al. [Michel et al., 2011] est l'un des meilleurs travaux pour la comparaison de deux contours des blobs et ensuite faire la correspondance entre un contour fermé et un autre ouvert. L'algorithme utilise en partie une version modifié de l'algorithme de déformation temporelle dynamique DTW [5].

5.5 L'utilisation des bibliothèques de simulation physique

Une approche utilisée pour ajouter un effet de manipulation naturelle avec les objets consiste à court-circuiter l'aspect de reconnaissance en entier. Au lieu de modéliser la main, ensuite reconnaître ses mouvements et sa posture en présence d'objets, on branche des bibliothèques de simulation physique à un modèle de la main qui n'est qu'un ensemble de sous particules ayant des propriétés physiques. Une scène avec des objets et un modèle de la main peut à première vue sembler être naturel et fonctionnel, mais on perd tous les aspects abstraits sur le fonctionnement interne de la reconnaissance ou les méta-informations sur les caractères de l'opération. Dans quelques travaux [Wilson et al., 2008, Hilliges et al., 2009] une bibliothèque physique a été ajoutée pour gérer les manipulations sur les objets.

5. http ://fr.wikipedia.org/wiki/Déformation_temporelle_dynamique

5.6 Conclusion

Dans ce chapitre, on a présenté les différentes techniques pour détecter des mouvements issus du corps humain à travers des capteurs et des périphériques d'entrée. On a ensuite présenté les algorithmes les plus utilisés soit pour extraire les informations utiles comme les tracés des données brutes de certains capteurs, soit pour reconnaître les formes de ces tracés et ainsi reconnaître les gestes.

Troisième partie

Contributions et développements

Chapitre 6

Extension du domaine gestuel

6.1 Introduction

Dans les premiers chapitres de la thèse, nous avons présenté un aperçu des travaux majeurs qui ont été effectués dans le domaine gestuel que ce soit dans la reconnaissance technique, ou d'un point de vue général dans la taxonomie des gestes et des manipulations. La divergence dans les approches permettant de voir un geste et de le traiter est un élément de richesse mais il ne peut être valorisé que si on arrive à organiser ces travaux suivant une ligne conductrice principale qui ordonne ces travaux dans un contexte plus large.

Dans ce chapitre, on va utiliser le modèle d'action de Donald Norman pour placer les briques fonctionnelles d'un geste. Ceci inclut bien sûr la manipulation réalisée sur un objet qu'on va appeler le geste apparent. Un geste apparent ou une manipulation apparente est ce qu'on voit se faire sur un objet. C'est peut être la partie "observable" puisqu'elle cache tout le processus cognitif qui a précédé l'opération. En mettant en évidence un cycle complet de la génération du geste, on constate que les travaux dans la partie qui précède le geste apparent ne sont pas connectés avec les travaux gestuels, ou il manque le point de vue d'interaction.

6.2 Définitions du geste et de la manipulation

On élargit la définition du geste en incluant tout le processus cognitif qui a été généré dans le cerveau pour choisir une opération à réaliser avec les mains ainsi que les déplacements dans le bras et la configuration ou la posture de main pour finir une opération. La partie finale, là où la main est déjà bien placée au dessus d'un objet pour le manipuler et jusqu'à la fin de la manipulation, est appelée "le geste apparent".

Une manipulation se distingue d'un geste par la présence d'un objet à manipuler. Elle peut être considérée comme une sous-partie du geste total qui a une durée de vie commençant dans le cerveau et se poursuivant jusqu'à la consommation de l'événement dans les applications cibles.

6.3 Les éléments d'une interaction gestuelle

L'une des premières études les plus reconnus sur l'interaction humain-machine est celle de [Norman, 2002] appelée le cycle de l'interaction humaine. Cette étude psychologique, définit les aspects présents dans une interaction en la séparant sur trois phases. La première phase est celle de la formation du but à partir de ce qui existe (1). La deuxième phase est celle d'exécution qui consiste à la transformation de ce but formé vers des tâches nécessaires non ordonnées (2), l'ordonnancement de ces tâches en une séquence (3), ensuite l'exécution de cette séquence (4). La dernière phase est celle de l'évaluation, on vérifie les résultats après l'exécution (5), on interprète la différence entre le résultat voulu et le résultat reçu (6), et on termine avec la comparaison des résultats (7).

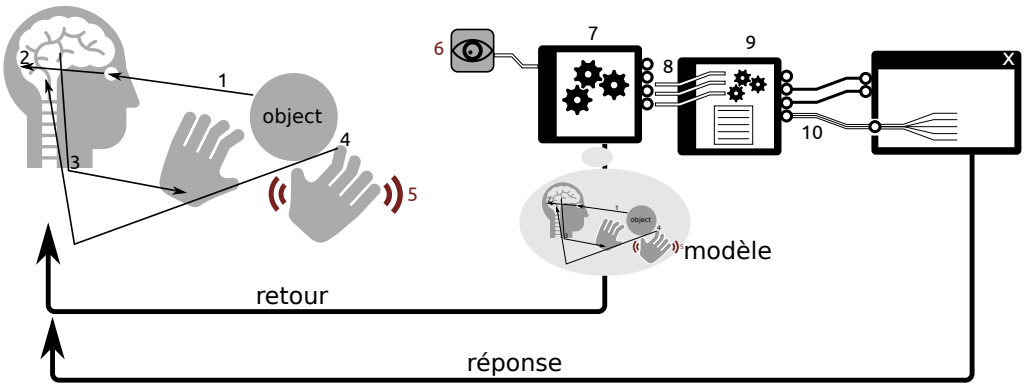
Dans la majorité des études IHM, la première étape et une partie de la deuxième sont généralement sous développées. La majorité des études s'intéressent principalement au suivi de l'exécution d'une tâche, et à partir des résultats, essayer de faire, d'après mon point de vue, une ingénierie

inverse pour estimer et conclure sur les meilleurs gestes à utiliser dans un contexte donné. Cette dernière approche est aussi plus sûre en terme de résultats et tests utilisateurs. On est sûr qu'on a dès le départ une base de gestes, et à la fin il suffit par exemple de faire les études statistiques sur les gestes les plus utilisés par les utilisateurs.

On a choisi de tester l'approche inverse, qui est la plus risquée, et d'essayer d'élargir le domaine des études de l'état de l'art pour arriver à trouver le lien à priori, avant l'exécution d'un geste. On vise pouvoir estimer, à minima, les intentions gestuelles de l'utilisateur. Il est possible de réaliser une correspondance entre le modèle de Norman et les briques des autres modèles cités précédemment. Pour une manipulation gestuelle, on commence par l'étape cognitive et vision pour la découverte de l'élément à manipuler, et c'est la formation du but chez Norman. L'étape de transformation du but vers des tâches non ordonnées, leur ordonnancement et l'exécution de cette séquence, correspond d'un coté à l'approchement (reaching) ainsi qu'au modèle de gestèmes de Signoret et North composé en une série de kinèmes.

Dans une interaction gestuelle avec des objets virtuels, les éléments qui influent sur cette interaction sont la main, l'objet lui même avec ses caractéristiques, mais aussi l'objectif de l'opérateur via l'interaction. C'est un processus différent qu'une interaction avec des objets tangibles tel que cité dans le travail de Wimmer [Wimmer, 2011] où on trouve la relation avec l'objet. Les autres aspects comme l'anatomie et quelques propriétés ne peuvent pas être les mêmes.

La figure 6.1 présente notre vision du cycle d'une interaction gestuelle tenant compte de l'humain et des outils informatiques. On commence par la découverte de l'objet à manipuler par la vision (1), réfléchir ce qu'il faut faire avec l'objet avec les éléments de mémoire appris pour gérer une situation similaire (2), Programmer les mouvements à faire pour atteindre l'objet et le rapprochement (3), La saisie de l'objet sur les plans d'opposition (4), Manipuler l'objet avec un geste apparent (5) et ici finit la partie lié à l'humain. La



1. Découverte avec le système de vision
2. Compréhension + proprioperception
3. Programmation des mouvements
4. Toucher et saisie
5. Manipulation : Geste apparent
6. Détection
7. Reconnaissance
8. Correspondance
9. Transformation
10. Routage

FIGURE 6.1 : Le cycle d'une interaction gestuelle

sixième étape concerne l'outil de capture utilisé (6), ensuite les algorithmes utilisés pour détecter les éléments clés et qui devraient prendre en compte le modèle de l'humain (7) pour ensuite générer des événements dans un système informatique prêts à être utilisés. Les étapes qui suivent font plutôt partie de la partie génie logiciel d'une interaction IHM, il y a l'étape de correspondance entre les événements gestuels et l'application cible (8), il y a aussi une étape optionnelle de transformation des événements vers d'autres plus compréhensibles par l'application (9), et finalement la dernière étape est la consommation des événements par l'application après leurs routage dans le composant interne cible (10).

C'est notre vision d'une interaction gestuelle dans un système IHM étendu regroupant différents domaines. Le geste apparent de manipulation ou visible peut être classé dans l'un des quatre cas de ce qu'on peut faire avec l'objet. Et avant que ce geste de manipulation apparent ne touche l'objet, il y a le mouvement de la posture de la main qui se forme. En analysant les mouvement du bras et de la main qui précède le geste, on formule l'hypothèse que l'on peut prédire une partie des intentions cognitives de l'utilisateur dans un contexte donné.

6.4 Naturalité et apprentissage

La naturalité du geste et le processus d'apprentissage sont deux concepts liés l'un à l'autre. Les gestes qui semblent naturels et utilisés sur les téléphones portables ont reçu une publicité avec des vidéos de personnes en train de les réaliser. La question sera de savoir si ces manipulations peuvent être réalisées sans que des sujets voient les vidéos. Est-ce que leurs interactions avec l'interface seront les mêmes ? Il existe deux modèles pour la compréhension et l'apprentissage. Le premier modèle appelé "énaction" signifie l'apprentissage par la pratique, similaire à la façon avec laquelle on apprend à conduire un vélo, en pratiquant le processus jusqu'à le maîtriser. L'autre modèle appelé en anglais "Vicarious/Observational learning" est l'apprentissage en observant d'autres usagers faire le geste, et ensuite les imitant. Les deux modèles sont valides et sont deux façons pour apprendre à manipuler un objet, qui ne s'arrêtent pas aux êtres humains.

La naturalité est un concept beaucoup plus complexe à traiter. La technique de scan fMRI pourrait donner une idée sur les activités résultant d'une fatigue et qui est en relation avec la naturalité. Le fMRI permet de mesurer l'activité cérébrale à travers le changement dans le flux du sang. Par rapport à l'état de l'art et à ces études, les types de saisie diffèrent dans le traitement par le cerveau. Les saisies de précision génèrent plus d'activité dans le cerveau et donc plus de stress. Alors qu'une saisie de force, bien qu'elle utilise plus de force musculaire, utilise généralement une seule moitié du cortex. Une activité naturelle et qui a été introduite par le long processus de l'évolution ne devrait pas causer du stress et devrait avoir une morphologie et une activité cérébrale assez minimale.

L'affordance d'un objet est définie par son pouvoir à évoquer son utilisation. En première approximation, on peut définir l'affordance gestuelle de manipulation par la capacité d'un objet à inspirer son maniement.

6.5 Positionnement par rapport aux études gestuelles

Par rapport aux autres travaux, le but de la thèse est de trouver un moyen de lier les différents domaines de recherches autour de la génération du geste et les inclure dans l'étude. Même si les travaux ne citent pas directement le geste ou ne sont pas dans le domaine de recherche direct de l'interaction humain-machine, on pense qu'ils sont importants dans les prochaines études pour sortir du cadre actuel des recherches.

Le deuxième chapitre de l'état de l'art regroupe les travaux menés autour du domaine gestuel élargi. Les travaux sur la saisie d'objets sont les plus proches du geste et éventuellement plus évidents à exploiter pour aller un pas plus loin dans l'étude de ce qui se passe juste avant le geste.

Le travail de Chris Harrison [Harrison et al., 2014] s'inspire de la façon avec laquelle on saisit un objet, pour ensuite utiliser une projection de cette saisie sur une surface 2D et activer une interface selon l'emplacement de cette projection. L'idée précédente qui est déjà inspirée d'un travail de Wilson [Wilson, 2009] simulant une saisie sur table, fait un lien entre un geste et la saisie d'objet à l'instant d'interaction de l'interface. Les deux idées étudient la saisie mais sans capturer ce qu'était la forme de la main quand elle était en train de s'approcher de la surface.

Le travail de Halla Olafsdottir et al. [Olafsdottir et al., 2014] est à notre connaissance l'un des premiers à aller dans la direction de recherche des études des postures qui se forment avant le geste final sur l'objet. Dans son travail, elle a étudié les planifications des manipulations avec un point de vue IHM. Car la façon avec laquelle on saisit un objet a un effet sur ce qu'on va faire avec. Le suivi des postures pré-geste apparent nous donne des informations sur sa finalité. Avec sa méthode expérimentale sur une surface tactile, elle a testé les manipulations de translations seules, rotations seules, et un mélange de rotations-translations d'objets.

Les résultats qu'elle a trouvés montrent que les utilisateurs planifient au-

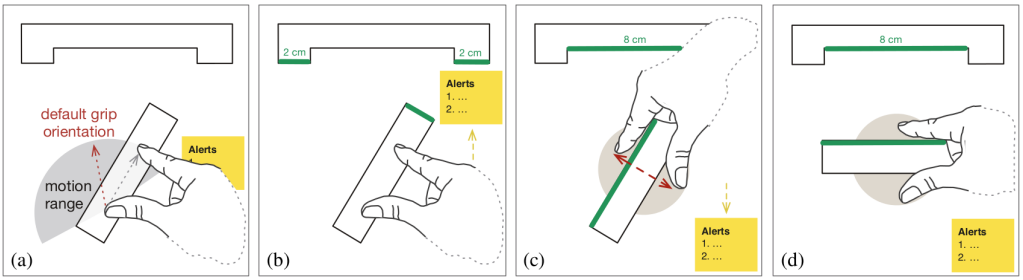


FIGURE 6.2 : L'impact de la posture initiale de la main sur le geste final à réaliser, l'utilisateur évite la rotation dans le sens anti-horaire et utilisation.

tant que possible l'orientation de leurs saisie pour la manipulation d'objets virtuels. Dans les autres cas, ils utilisent une posture générique. L'ouverture de l'IHM sur les études de la psychologie de saisie d'objets permettent de prévoir des interactions mieux adaptées. Dans la figure 6.2 Le système anticipe les mouvements de l'utilisateur en éloignant le carré jaune d'alerte loin du chemin prédit pour ne pas risquer l'occultation. Notre thèse essaye de tirer profit des études précédentes mais en utilisant des périphériques et moyens de captures en 3D.

6.6 Simplification de la taxonomie des gestes de manipulation

Dans un geste de manipulation, on interagit et manipule forcément un objet. Donc dans un tel contexte, on pense que l'inclusion de l'objet dans une taxonomie gestuelle de manipulation peut simplifier le processus. La main et l'objet sont les éléments les plus importants. L'objet lui ne peut supporter que quelques états définis. Il peut soit être touché, subir une rotation, une translation, et une déformation de sa forme. En observant les états possible de manipulation d'un objet, on peut proposer une famille de classes gestuelles de manipulation des objets comme dans la figure 6.3. Ces familles gestuelles définissent les gestes de façon globale, si on veut reconnaître un geste spécifique plus fin, il faut ajouter un autre niveau de classification

selon les propriétés respectives de chacune des classes principales. Le geste "swipe" est donc un geste "Drag" avec le paramètre vitesse qui est plus grand que la normale.

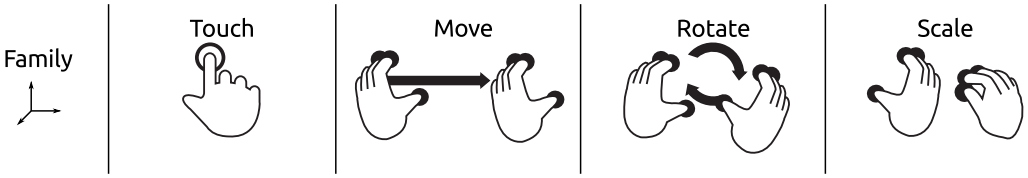


FIGURE 6.3 : Les familles principales de gestes 3D

L'artiste Gabriele Meldaikyt [1] a ré-imaginé les gestes multi-tactiles appliqués sur des objets réels. Dans la figure 6.4 il ré-implémente les gestes d'agrandissement, de mouvement et toucher sur des objets faits en bois et cartons. La vision de l'artiste réduit la séparation entre les gestes 2D et ceux réalisés en 3D.

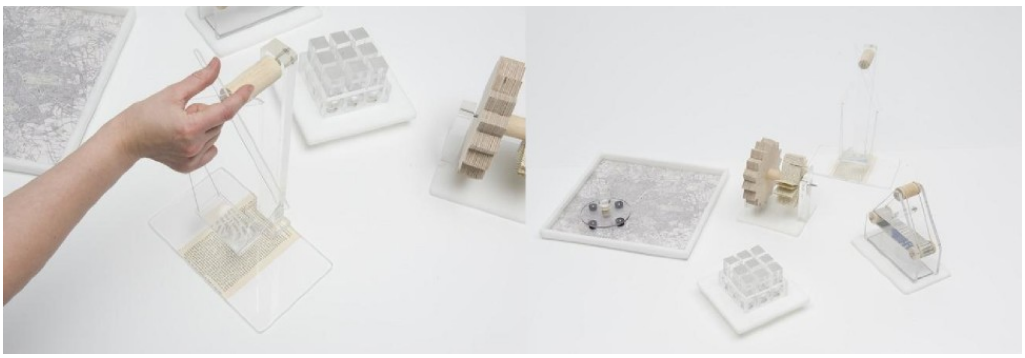


FIGURE 6.4 : Les gestes mutli-tactiles représentés avec des objets 3D par Gabriele Meldaikyte

Les gestes effectués actuellement en 3D diffèrent de ceux réalisés sur une surface. Sur une surface, on sait exactement quand est-ce qu'un geste apparent commence et quand est-ce qu'il se termine. Le déterminisme vient du fait que la zone d'interaction est une surface limitée. En 3D, l'ajout de la nouvelle dimension n'a pas été faite en créant une boite avec une surface

1. http ://www.gabymel.com/multi-touch-gestures/

d'entrée dans chaque face. Même si de telles solutions ont existé, les gestes en 3D n'ont pas une surface depuis laquelle on sait qu'ils commencent à être générés. Une des solutions à ce problème réside dans l'étude du geste étendu et estimer depuis la posture de la main, quand est-ce qu'une manipulation avec un objet devraient être prise en compte.

Pour résoudre ce problème, il faut détecter de façon algorithmique le commencement et la fin d'un geste, et il existe plusieurs solutions à ceci. L'une des solutions actuelles est de spécifier des postures spécifiques de la main pour commencer un geste et pour le finir, ou utiliser une seule posture le temps de la réalisation. L'autre solution s'inspirant de la reconnaissance vocale, est de suivre les mouvements de la main, on fixe la main dans un emplacement pour dire qu'il faut commencer la reconnaissance, la bouger, et ensuite fixer sa position une autre fois pour dire qu'on a terminé.

6.7 Prédiction des manipulations gestuelles

Après l'étude de l'état de l'art, et en ayant conscience que la manipulation gestuelle est un processus qui ne se limite pas à l'instant où on touche les objets, on propose de focaliser sur la partie qui se passe exactement avant la manipulation. En allant un pas vers l'arrière dans l'échelle temporelle, on propose de vérifier la posture de la main avant de saisir un objet.

Un sujet, avant de manipuler un objet, doit s'approcher de cet objet ensuite préparer la posture de sa main pour le saisir. C'est dans cette préparation qu'on peut avoir des informations sur le but de l'utilisateur. Pour qu'un utilisateur saisisse un cube, la notion de plan d'opposition intervient et une forme de pince se crée en fonction de la position du cube et son orientation.

Si l'utilisateur voulait toucher, déplacer ou déformer l'objet, la posture de la main serait différente et elle commencerait à se former bien avant que la main atteigne l'objet en question.

La figure 6.5 montre 4 cas de prédictions de la forme de la main avant

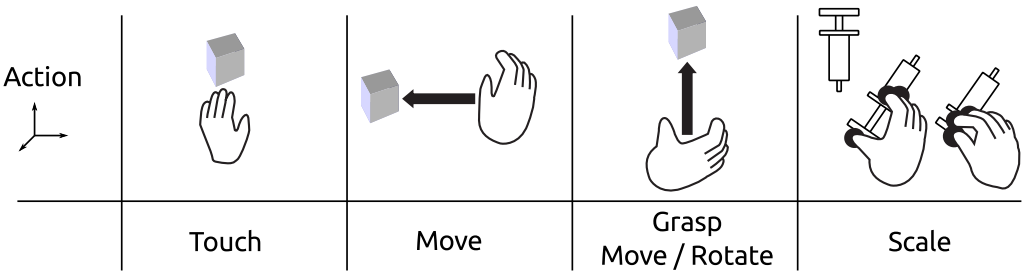


FIGURE 6.5 : La prédiction de la manipulation gestuelle

la réalisation de la manipulation. Cette prédiction nécessite l'accès à l'information de capture de la main et non pas une donnée pré-traitée depuis un algorithme auquel on n'a pas accès. La reconstruction de toute la chaîne de reconnaissance et de manipulation demeure nécessaire.

6.8 Les choix des technologies de capture

Dans notre contexte, on voulait détecter une large zone d'une table. Le périphérique de détection devrait aussi être mis dans un emplacement qui ne dérange pas les opérations. Après avoir fait le point sur les choix technologiques, tous les périphériques qui ont une petite zone de capture ne sont pas adaptés à nos besoins, donc à éliminer. Le choix reste entre les caméras RGB, et les périphériques de captures 3D qui ont tous une large zone de capture.

Dans l'installation déjà présente dans le laboratoire, un projecteur est utilisé pour projeter sur la table. Les variations de couleurs projetées peuvent facilement nuire à l'entrée de la caméra RGB et ses algorithmes de suivi de la main ou du bras. Les algorithmes les plus faciles à utiliser sont ceux basés sur la couleur de la peau humaine, et une projection peut avoir des couleurs qui en sont proches.

Parmi les choix qui existaient en début de thèse, il y avait la première version de la Kinect. Elle permet de capturer une large zone et il est possible de l'installer dans un emplacement qui ne gène pas les utilisateurs au dessus

de la table.

Dans notre étude, on a aussi besoin de plusieurs paramètres d'un processus gestuel qui ne sont pas fournis par un périphérique qui traitent les données en interne et ne fournit que des résultats finaux. On a besoin d'accéder à la capture des données brutes pour y suivre et capturer par exemple les détails de la saisie et tout ce qui se passe avant la manipulation gestuelle de manipulation d'objets. La Kinect permet de fournir une entrée brute sous forme d'un nuage de points, ou plus précisément une surface de points 3D. Il reste donc à reconstruire la chaîne de reconnaissance qui peut exister dans un périphérique comme la Leapmotion, mais en développant nos propres algorithmes. La forme de la main avant le geste apparent induit l'opération que l'utilisateur va faire avec cet objet. Il faut donc accéder à la posture 3D de la main tout au long d'une opération, et qui est une chose non disponible via la majorité des capteurs autre que la Kinect au moment du lancement de la thèse.

6.9 Conclusion

Dans ce chapitre, on a présenté les principaux problèmes qui caractérisent un geste 3D par rapport à celui qui est effectué sur les tables tactiles. On a aussi posé des questions sur la source des interactions et manipulations naturelles. On a essayé de renforcer le lien entre les chapitres de l'état de l'art et la ligne directrice des travaux de la thèse. La piste de la prédiction des manipulations à partir de la forme de la main avant atteindre l'objet est une piste à explorer, et qui nécessite l'utilisation des choix techniques argumentés dans le dernier paragraphe.

Chapitre 7

Système de reconnaissance développé

7.1 Introduction

Après avoir discuté les aspects théoriques des travaux autour du geste et après avoir présenté les techniques permettant sa détection et sa reconnaissance, on présente dans ce chapitre le système construit pour faire les reconnaissances gestuelles et l'enregistrement de l'activité utilisateur.

7.2 Description du système et des choix

Notre but principal était de réaliser un mécanisme permettant de reconnaître les gestes sur table et dans l'espace au dessus de cette table. La reconnaissance se situe dans l'espace de proche de l'utilisateur, mais peut avoir un effet sur l'espace moyennement proche ou lointain. Les technologies disponibles au moment de la réalisation de la spécification technique n'étaient pas très nombreuses, on avait le choix entre continuer à utiliser une ou plusieurs caméras RGB, par exemple la Sony PS3Eye, soit utiliser la Kinect qui venait d'être lancée. On avait déjà un système de projection utilisant un vidéoprojecteur et un miroir, les couleurs projetées sur la surface

de la table sont aussi projetées sur la surface des mains. Un suivi qui utilise une caméra 3D n'est pas altéré par le changement de la couleur puisqu'il utilise la bande de l'infrarouge.

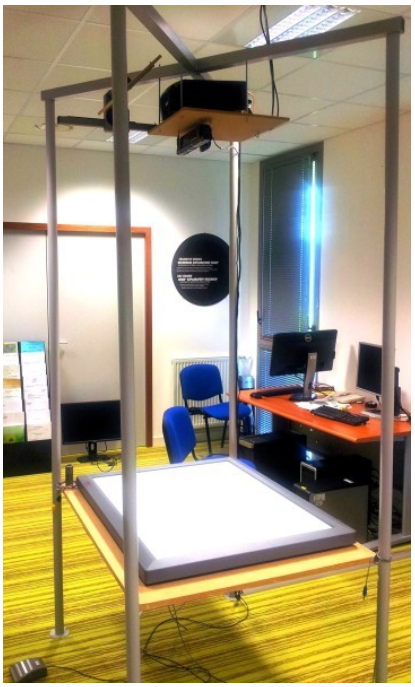


FIGURE 7.1 : L'installation de la Kinect

Notre choix s'est porté sur la Kinect qui permet de suivre une zone assez large, comme indiqué dans le quatrième chapitre sur la comparaison des périphériques, et qui n'est pas perturbée par une projection de couleurs sur une cible. Elle a été installée au plafond presque au même niveau que le vidéoprojecteur et elle est orientée vers le bas pour capturer la zone de manipulation située au dessus de la table et fournir les points en 3D. Le périphérique Asus Xtion aurait pu remplacer la Kinect par la suite puisqu'il contient le même module infrarouge mais ne nécessite pas une alimentation et il est plus léger d'après les analyses du chapitre sur l'état de l'art technologique.

Pour accéder aux informations de capture fournies par ce périphérique, on avait le choix entre le SDK de Microsoft, le SDK OpenNI et la biblio-

thèque libfreenect. Cette dernière bibliothèque était la première à fournir un accès au périphérique au moment où elle était destinée aux consoles de jeu. La bibliothèque fournit un accès bas niveau au nuage de points 3D sans utiliser trop de mémoire et alourdir les traitements. Elle envoie la séquence d'initialisation pour démarrer la Kinect puis commence à recevoir les données binaires du capteur. Elle transforme ces données reçues de façon synchronisés vers une matrice OpenCV avec une taille de 640x480 points. Il n'existe pas de module pour des traitements plus complexes.

Le SDK de Microsoft et OpenNI sont concentrés sur une utilisation standard avec le capteur en face d'un utilisateur. Il permettent la capture du squelette du corps de l'utilisateur et fournissent des méthodes d'accès à une partie dans le corps avec une synchronisation avec le nuage de points qui va avec. Il faut noter qu'on n'a pas besoin du squelette pour notre application puisque elle n'est pas un cas standard d'utilisation supportée par ces bibliothèques.

La bibliothèque libfreenect est donc la plus simple à utiliser pour accéder au nuage de points de la Kinect. Par contre, on a donc toute la chaîne de reconnaissance de gestes à construire et à maîtriser pour avoir un résultat similaire et applicable sur notre cas.

7.3 Système de suivi

7.3.1 Gestion du flux de la Kinect

À ce stade, on a un nuage de points 3D comme entrée de base de notre système. Il faut utiliser les algorithmes de traitement d'image pour gérer ce flux en temps réel. Dans [Boulabiar et al., 2011], on a fait un système capable de détecter et suivre la main en utilisant une caméra et se concentrant sur la couleur de la peau.

Ce dernier système utilise une autre forme de flux qui est le flux vidéo, et d'autres algorithmes pour segmenter la zone d'intérêt dans les images, mais il est possible de s'en inspirer pour arriver à des résultats similaires.

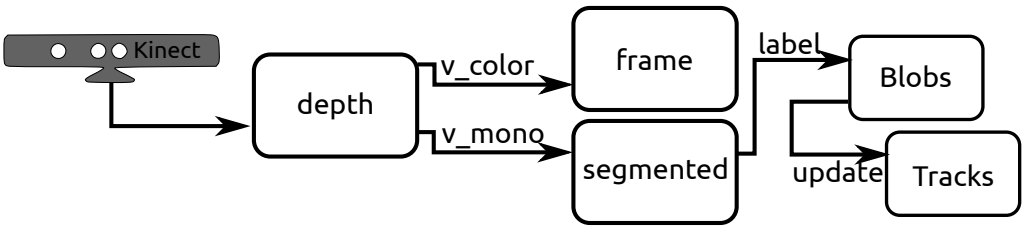


FIGURE 7.2 : Traitement du flux de données reçus depuis la Kinect

Le nuage de points 3D de la Kinect n'est réellement qu'une seule surface 3D. La Kinect et tous les périphériques cités précédemment ne peuvent voir ce qui est occulté. Cette limitation matérielle nous a conduit à ne pas trop se focaliser sur l'aspect 3D pour des opérations de suivi de la main qui est une opération d'identification. On utilise les informations 3D dans l'étape suivante une fois la main suivie. À travers la figure 7.2 on explique comment on gère ce flux. On reçoit le flux dans une structure de donnée matricielle appelée "depth" dans le diagramme, elle contient les valeurs des points en 3D, et chaque point est une valeur codée sur 16bits. Cettre structure change de valeur 30 fois par seconde et remplie par la fonction $freenect\_sync\_get\_depth\_cv()$

On construit avec la fonction *view_color* la structure "frame" qui est une image RGB sous forme d'une matrice de 3 couches contenant chacune 640 x 480 valeurs, et dont chaque valeur est codée sur 8bit. Cette structure sera utilisée pour voir la scène captée sous forme de couleurs. La fonction *view_color* qu'on a codée transforme les valeurs codées sur une seule couche de 16bit initiale, vers une proportion qui remplira ensuite un espace de couleur HSL. Ces valeurs seront utilisées pour changer de l'espace de couleurs HSL vers celui RGB requis pour les fenêtres d'affichage OpenCV. La décision d'utiliser un espace de couleurs intermédiaire est prise car la transformation des données de profondeurs vers des données de couleurs est linéaire et simple, ce qui n'est pas le cas si on voulait faire une transformation directe vers RGB.

La fonction *view_mono* transforme les données de profondeurs de "dep-

th" vers une matrice d'une seule couche binaire appelée "segmented". Elle sera utilisée pour appliquer les algorithmes de marquage de blobs à travers la fonction *label*, et le suivi dans le temps à travers la fonction *update_tracks*. Ces deux fonctions seront expliquées dans les sections suivantes. Nous avons donc aplati le flux de la Kinect pour avoir une image similaire à ce que nous avons obtenu dans le projet précédent de suivi avec la caméra. L'aplatissement consiste à ignorer la 3e dimension et à copier le flux dans une structure image OpenCV.

7.3.2 Algorithme de marquage des blobs

Avec ce choix on obtient la nuée de points nommée blob qui rentre dans la zone, on peut alors utiliser des algorithmes d'imagerie 2D maintenant possibles à utiliser. Ces algorithmes de marquage et de suivi 2D ont fait l'objet de nombreux travaux de recherches et ils sont bien optimisés pour un traitement temps-réel. L'algorithme utilisé à travers la bibliothèque cvblob avec OpenCV est celui développé par Chang et al. [Chang et al., 2004] pour le marquage des blobs en utilisant un seul passage à travers les lignes de l'image.

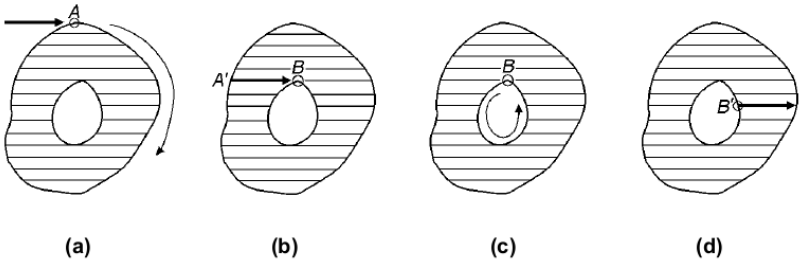


FIGURE 7.3 : Les quatre étapes d'étiquetage des blobs

L'algorithme procède en 4 étapes principales d'un parcours du haut vers le bas et de droite à gauche de l'image :

1. Si on trouve un point extérieur pour la première fois, on trace le contour jusqu'à revenir à A et on l'étiquette.

2. À chaque un pixel déjà étiqueté, on suit la ligne pour étiqueter en A' les pixels équivalents.
3. Si on trouve un premier pixel du contour intérieur, on trace tout ce contour là.
4. À chaque pixel du contour intérieur, on scanne et on étiquette les pixels à droite. De cette façon, les pixels intérieurs sont visités une seule fois, et ceux du contour au maximum 4 fois.

7.3.3 Algorithme de suivi des blobs

En utilisant le dernier algorithme, on a maintenant des zones marquées comme étant objets à travers une image. Il reste à pouvoir suivre les mouvement de ces zones ou blobs temporellement d'une image à un instant t à t+t0 sans perdre le marquage. Pour faire ceci on se base sur l'algorithme défini par Senior et al. [Senior et al., 2006] permettant le suivi sans perdre l'objet avec une simple occultation. Cet algorithme de suivi utilisé permet de grouper les objets après les avoir décrits utilisant un rectangle entourant et un masque d'image de la zone importante dans ce rectangle. À travers la succession des images (et du temps), le processus associe chaque région à un objet suivi "Track" parmi ceux qui sont déjà initialisés. Une matrice des distances est construite en calculant la distance entre la région non encore attribuée et la liste des objets suivis active "Tracks". La distance utilisée est une distance entre les boîtes englobantes du centre du premier rectangle au point le plus proche de l'autre rectangle comme indiqué sur la figure 7.4. Si les 2 centres sont dans le même rectangle, la distance est nulle. Une deuxième distance temporelle est ajoutée pour pénaliser les objets traqués qui ne sont plus visibles pour une certaine durée de temps La matrice des distances est transformée en une matrice de correspondance après seuillage des valeurs et leurs transformations en 0 ou 1. La résolution du suivi est ensuite traitée selon 4 cas de figures plus détaillées dans la référence.

Pour résumer, on a donc plusieurs phases, la détection des zones 3D dynamiques dans la scène et qui représentent la main, le coude et le bras,

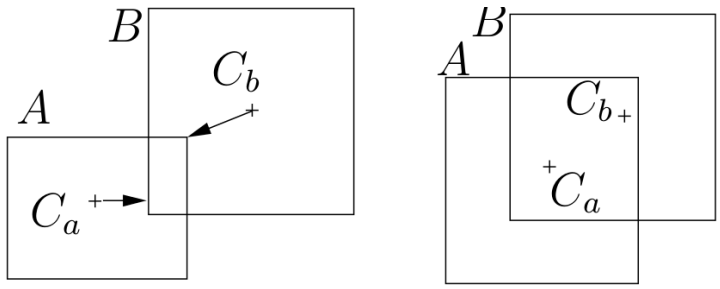


FIGURE 7.4 : Calcul de la distance entre les boites englobantes de deux images, dans 2 cas différents [Senior et al., 2006]

et ensuite le suivi de ces zones afin de ne pas les perdre au cours du temps. On a utilisé les algorithmes de marquage à une seule passe par image pour étiqueter les zones dynamiques dans la scène, et ensuite un algorithme similaire pour ne pas les perdre. Ce choix nous a permis de suivre les blobs et ne pas les perdre en leur affectant des identifiants. Les calculs se font en temps-réel et s'exécutent plus rapidement que les algorithmes de suivi de nuage de points en 3D.

Par contre, le blob suivi contient à la fois la main de l'utilisateur et le reste de son bras comme indiqué dans la figure 7.5. On s'intéresse seulement à la main pour l'instant, il faut donc extraire juste la zone correspondante à la main.

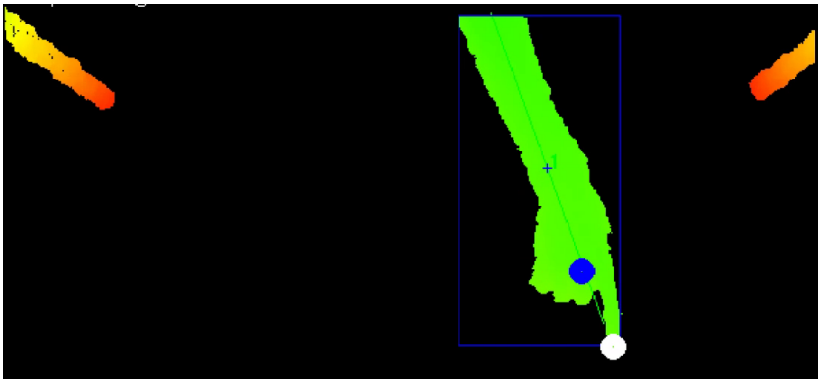


FIGURE 7.5 : Visualisation du blob contenant la main et le bras

7.4 Système de détection de la main

Dans notre installation, l'opérateur qui interagit avec la table fait rentrer ses mains à l'intérieur de la zone de suivi, et c'est à ce moment là que le système commence le suivi de cet objet. L'objet dans ce contexte est appelé blob, et le terme signifie simplement un objet suivi de façon abstraite sans encore lui donner une signification fonctionnelle. Avec les contraintes temps-réel, on devrait seulement trouver des algorithmes efficaces pour détecter la main dans notre cas. On a donc cherché à trouver des heuristiques.

Notre système de détection de la main utilise une heuristique simple, dans une manipulation, l'utilisateur tend sa main, donc on cherche la direction de la main et du bras. Cette direction peut être détectée en cherchant la plus longue ligne qu'on peut dessiner à l'intérieur du blob. Le contour du blob a été simplifié pour ne contenir que les points les plus importants. La détection de la plus longue ligne ne prend que très peu de temps.

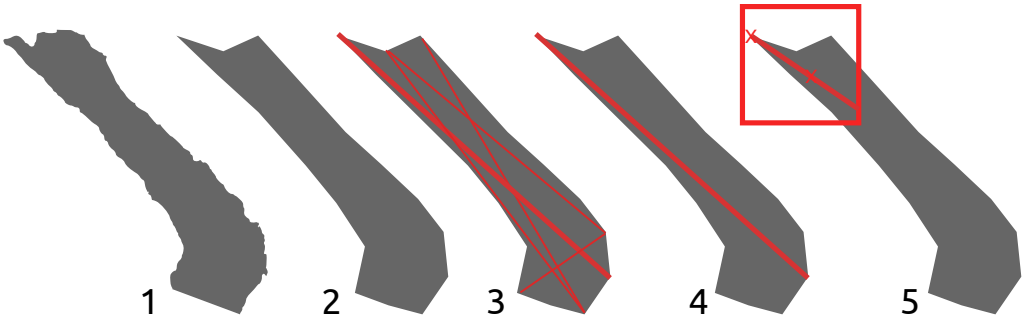


FIGURE 7.6 : Étapes pour la détection de la main

La plus longue ligne permet de détecter le doigt pointeur ou l'extrémité de la main, et la source du bras. Le doigt pointeur est celui qui se rapproche le plus du centre de la table. On arrive donc à détecter le doigts pointeur d'un ou plusieurs blobs (qui représentent un ou plusieurs mains et utilisateurs).

Pour extraire la main, on utilise une méthode qui consiste à couper suivant une distance prédéfinie et suivant la profondeur dans l'axe z. Les résultats expérimentaux ont montré que cette heuristique fonctionne bien et

permet d'extraire la main dans notre cas en temps-réel. On n'a pas besoin d'avoir des algorithmes complexes pour obtenir ce résultat.

FIGURE 7.7 : entrée et sortie du module de détection de la main

Des techniques pour améliorer l'extraction peuvent être proposées, mais ne sont pas réalisées pour l'instant, car ils ne sont pas prioritaires par rapport au développement d'algorithmes de reconnaissance du geste lui même. Une des méthodes d'amélioration est de détecter le poignet et couper à partir de là comme indiqué dans la figure 7.8.

Donc, à la sortie de ce bloc de traitements, on arrive à avoir deux points, celui du doigt pointeur, et celui du centre de la main. Les informations 3D de ces points sont obtenues en lisant depuis la structure de donnée contenant la donnée de la Kinect et qui est maintenue à disposition en cas de besoin, même si les autres algorithmes fonctionnent sur des données aplaties.

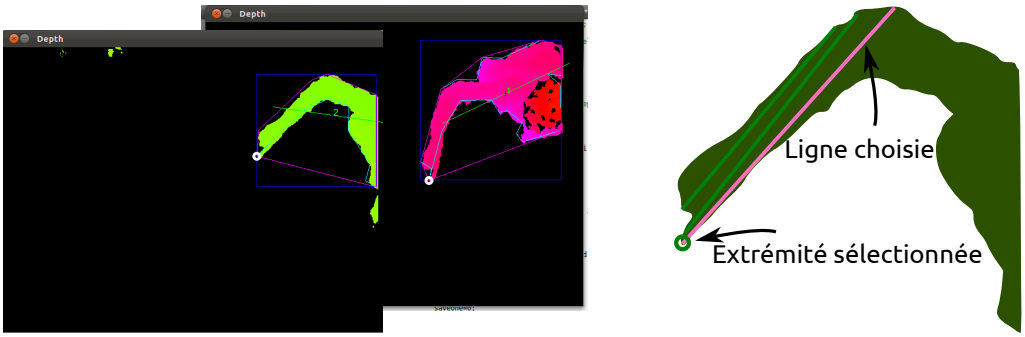


FIGURE 7.8 : Détection du doigt pointeur et de la main

7.5 Conclusion

Dans cette partie, on a fait les choix de base pour la reconnaissance des gestes tout en voulant maîtriser le cycle complet. Travailler avec un nuage de points semble assez bas niveau, mais on arrive à détecter assez rapidement la main et à exploiter le nuage de points associé dans les modules suivants de détection et de sauvegarde des informations.

Chapitre 8

Reconnaître les activités de la main

8.1 Introduction

Que ce soit en utilisant les périphériques de captures, les caméras couplées avec les algorithmes en imagerie, en “soft computing”, ou de reconnaissance des tracés, le but final est forcément de pouvoir reconnaître et suivre la main. Par la reconnaissance de la main, on signifie principalement extraire sa position dans l’espace, son contour, sa forme, ou la configuration des doigts. Et en couplant ces paramètres dans le temps, on aura techniquement les éléments pour ensuite extraire la signification d’un mouvement ou un autre.

Faisant partie d’un long cycle comme présenté dans les chapitres précédents, cette section correspond seulement à la partie sur la capture du geste et son interprétation technique. La reconnaissance de la configuration de la main est appelé la reconnaissance de la posture de la main ou du geste statique. La reconnaissance du mouvement dans le temps quant à lui est appelé geste (en anglais “gesture”), parfois se limitant seulement à l’interprétation du mouvement, et dans d’autres cas incluant l’identification de la posture pour différencier un geste par rapport à un autre, comme c’est le

cas dans le langage des sourds-muets.

8.2 Reconnaissance des postures

Pour arriver à reconnaître les postures, nous avons utilisé dans un précédent travail [Boulabiar et al., 2011] les informations du contour de la main. Avec un objectif de reconnaître un sous ensemble de postures de la main, il suffit de générer une base de modèles des postures spécifiques, et de vérifier à partir de l'entrée reçue à chaque fois si on a à un instant t le une posture similaire à ce qu'on a dans la base de référence. C'est en résumant un problème d'appariement de formes. L'enregistrement des modèles et la comparaison sont limités par les informations qu'on a pu détecter et extraire du signal brut.

En essayant de reproduire le travail effectué précédemment avec une caméra normale [Boulabiar et al., 2011], on a détecté la configuration de la main en utilisant seulement l'information de son contour. En premier lieu, on a utilisé une sortie fournie par la Kinect qui, d'après le chapitre précédent, arrive à reconnaître et extraire la main seule.

Une fois le blob de la main localisé, on l'isole de la zone totale de capture et on l'aplatit sans considérer la composante Z en 3D. Ce blob est ensuite transformé en une structure OpenCV 2D. On applique les algorithmes pour générer un contour de blob ayant la forme de la main.

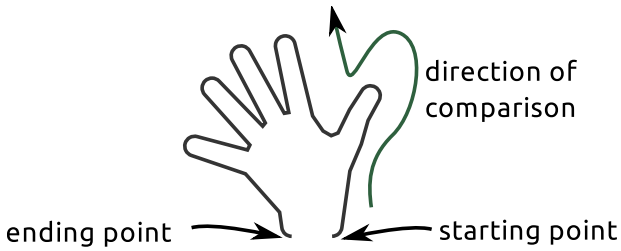


FIGURE 8.1 : Mécanisme de reconnaissance de la posture de la main en utilisant le contour

On enregistre plusieurs formes pour les contours de la main afin de les

utiliser comme une bibliothèque de modèles. Ensuite on compare le contour pris en temps réel avec celui que l'on a dans la bibliothèque. Pour faire la comparaison, on utilise l'algorithme 1$ appliqué au contour de la main après l'avoir traité comme un tracé comme indiqué dans la figure 8.1 qui normalise le contour à comparer et calcule la distance qui sépare ses points avec ceux qui sont dans la bibliothèque des modèles. Chaque comparaison calcule une distance qui représente la somme des erreurs.

On sélectionne la référence la plus proche au sens de la métrique utilisée, et on vérifie que celui-ci est supérieur à un certain seuil minimal. On peut dire donc qu'on a reconnu une certaine posture de la main. On a choisi dans un premier temps de se limiter aux contours de la main en 2D pour simplifier les algorithmes et réduire les erreurs. Il est à rappeler que l'entrée de la Kinect est une surface 3D dans la majorité des cas incomplète, et qui n'arrive pas à voir les points derrière cette surface.

8.3 Reconnaissance des gestes simples

Dans une approche pour reconnaître les gestes simples en 3D, on a travaillé sur l'enregistrement des déplacements du centre ou de l'extrémité de la main en 3D, et essayé de les classifier pour les reconnaître. On a pris encore une fois l'algorithme 1$ et on l'a modifié pour qu'il supporte la 3e dimension.

La modification a essentiellement porté sur la partie du calcul de la distance qui est transformé vers un calcul de la distance euclidienne en 3ème dimension. Sur la partie de la normalisation, le placement des points est repositionné dans un cube au lieu d'un carré, et finalement le ré-échantillonnage est fait en 3D.

En faisant des premiers tests avec une base de 4 gestes 3D dans l'espace, comme indiqué dans la figure 8.2, on arrive à les reconnaître une fois réalisés. Malheureusement, bien qu'on arrive à reconnaître ces gestes du point de vue technique, il s'avère pratiquement inutile dans un contexte applicatif

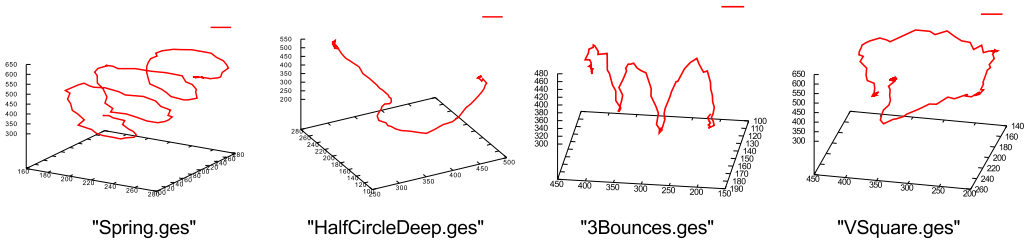


FIGURE 8.2 : Les 4 gestes 3D testés

de reconnaître des gestes 3D à base de tracés dans le but de contrôler un système. Nous avons demandés aux sujets de l'expérience de définir et ensuite de réaliser des gestes métaphoriques en 3D. Ils ont eu beaucoup de problèmes dans chacun des cas relatives à la mémorisation du geste métaphorique.

La définition d'un mouvement 3D dans l'espace est difficile à spécifier puisque c'est une action que les humains ne sont pas habitués à faire dans un contexte autre qu'un discours. L'humain est habitué à faire des tracés sur une feuille mais pas dans l'espace. Les opérateurs après avoir eu des difficultés pour définir des tracés simples pour déclencher certains actions, ont eu aussi des difficultés pour les reproduire. Les tracés 3D sont soit refaits en inversé, ou distordus.

La motivation pour continuer à développer cette piste a perdu l'intérêt puisque c'est inutilisable dans un cas pratique dans un système réel. Par contre, cette même piste prouve que les humains ne sont adaptés aux gestes métaphoriques surtout s'ils sont réalisés en 3D. L'utilisation des manipulations directes sur les objets est le chemin à suivre pour les interactions qui ne devront pas être mémorisés.

8.4 Enregistrement des informations

Les données du bloc précédent nous fournissent l'information sur le doigt pointeur et le centre de la main. On a aussi les structures de données du nuage de points qui sont maintenues. D'après ces données, on peut aussi

extraire le nuage 3D des points de la main.

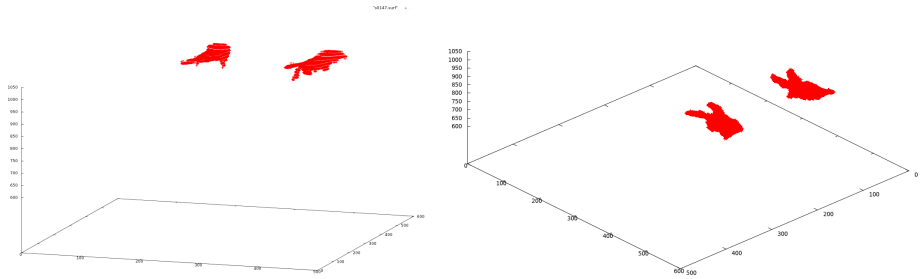


FIGURE 8.3 : Le nuage de point des main dans la scène

Le système d'enregistrement a pour but de sauvegarder les données pour une future utilisation, ou pour des besoins de débogage en visualisant la trajectoire de la main.

On a défini un système d'enregistrement de log qui permet de sauvegarder la position du centre de la main pendant une période choisie par l'opérateur. Les informations sont sauvegardées dans un fichier texte qui contient les coordonnées x, y, et z séparées par un espace. Ce fichier peut être lu avec Gnuplot.

Pour la main, une zone de 75 pixels autour du point central de la main est aussi sauvegardée à chaque frame, au lieu d'ajouter un point au fichier de sauvegarde de la trajectoire. Les fichiers contenant la forme de la main dans le temps contiennent chacun les coordonnées x, y et z de chaque point non vide dans le carré de 150 pixels 3D de coté. Ils peuvent eux aussi être visualisés avec le logiciel graphique Gnuplot.

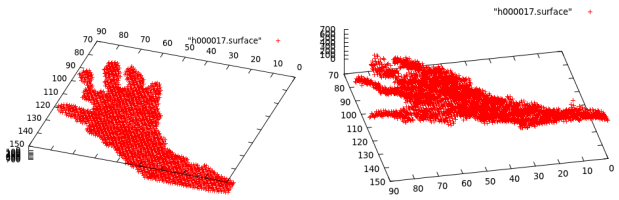


FIGURE 8.4 : Le nuage de point de la main

8.5 Pointage sur table

Comme notre installation sur table tactile est relativement grande, l'utilisateur n'a pas un accès facile à toutes les zones de la table. Un tel problème a été discuté dans le travail de Marquardt et al.[Marquardt et al., 2011] en citant l'espace de travail sur une table. On a donc pensé donner à l'utilisateur le moyen de pointer vers les zones distantes de la table.

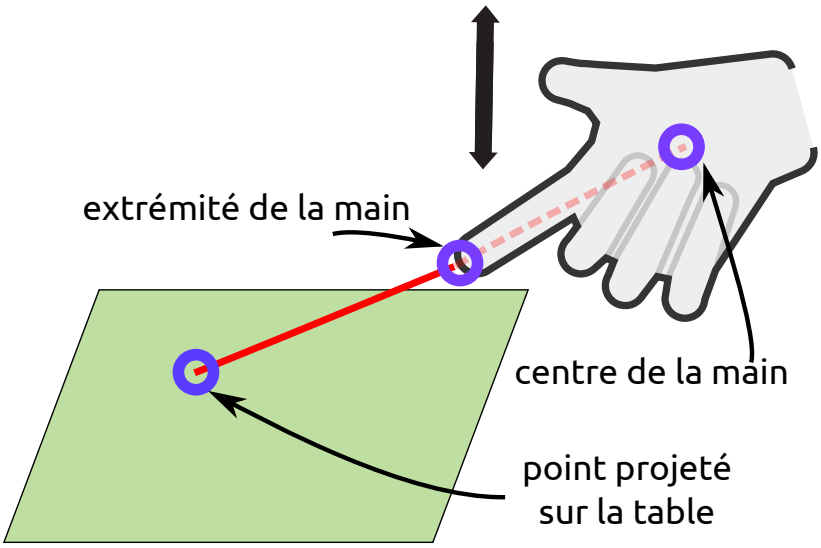


FIGURE 8.5 : Pointage sur la table avec la main

Le pointage se base comme représenté dans la figure 8.5 sur l'intersection de la direction du doigt et du plan de la table. Le plan s'étend sur la surface (x,y) et il a une composante z constante, considérée comme nulle pour la simplification des équations. La direction du doigt est définie par l'extrémité de la main, et par son centre. Le système d'équations permettant de résoudre le point d'intersection est donné en 8.1 avec P1 le point de l'extrémité de la main, P2 le centre de la main, et P le point d'intersection.

$$\begin{cases} 0 = Ax + By + Cz + D \\ P1 = (x1, y1, z1) \\ P2 = (x2, y2, z2) \\ P = P1 + u(P2 - P1) \\ u = \dfrac{(Ax1 + By1 + Cz1 + D)}{(A(x1 - x2) + B(y1 - y2) + C(z1 - z2))} \\ z = 0 \end{cases} \tag{8.1}$$

Vu que l'entrée brute de la Kinect est bruitée, et avec une certaine marge d'erreur, les positions du centre de la main ainsi que celui de l'index peuvent générer des positions qui sont très bruitées. Le point indiqué sur table bouge et saute d'une position à une autre malgré la stabilité de la main puisque un petit changement dans la direction de la main ou dans la position choisie de l'index dans le nuage de point sélectionnée est multiplié sur un point d'intersection lointain. La position du point d'intersection, peut arriver à une différence de 2 à 3 centimètres dans les extrémités de la table. Utiliser un dispositif plus précis peut améliorer ce résultat dans le futur. Pour résoudre ce problème on a utilisé un filtre simple qui calcule la moyenne des deux dernières positions du point. D'autres filtres ont été testés comme le 1 Euro par Géry Casiez et al. [Casiez et al., 2012], et leurs interface de comparaison des filtres nous a aidé à choisir. En simulant une entrée similaire à celle générée à la sortie de notre algorithme, on a trouvé qu'un filtre simple comme celui qu'on a choisi comme l'exponentiel simple fonctionne mieux avec les données bruités que l'on obtient sans l'effet de latence. L'ajout d'un filtre a aussi des inconvénients, on perd dans la réactivité du pointage, puisque il faut maintenant un cycle d'instruction supplémentaire à exécuter pour pointer un emplacement. Le point prend du temps pour s'afficher à la destination et l'exponentiel simple permet d'avoir le moins de latence.

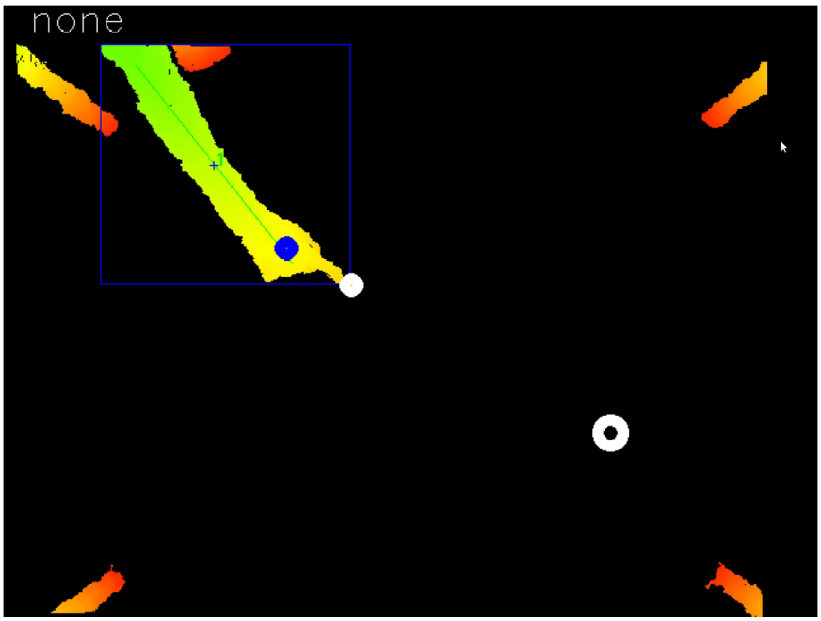


FIGURE 8.6 : Visualisation du point de pointage sur la table

8.6 Les autres modules de reconnaissance

8.6.1 Les couches d'interaction

Un geste ou une posture réalisée peut avoir des significations différentes suivant l'emplacement où ils ont été générés. Le travail de Marquardt et al. [Marquardt et al., 2011] présente quelques scénarios de manipulation 3D dans l'espace. La notion de prise en compte des couches d'interaction pour par exemple gérer la précision existe, mais sans qu'il y ait un système réel développé derrière.

Notre système arrive à générer des événements sur la présence de la main dans une couche supérieure. Par contre, c'est à l'utilisateur final de gérer la composition des messages des couches avec ceux de l'élévation.

8.6.2 La détection des clics

Avec un focus principal sur la détection de l'activité 3D de la main au dessus de la table, il est aussi nécessaire de détecter son positionnement sur

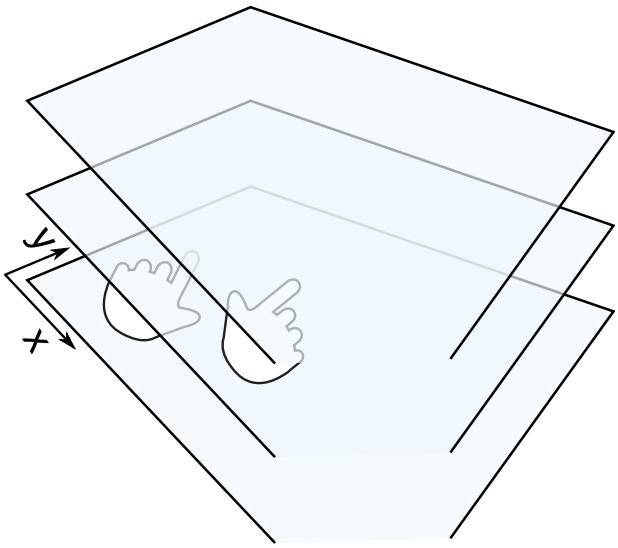


FIGURE 8.7 : Différentiation de la signification selon la couche d'altitude de manipulation

la surface de cette table. La détection du contact des doigts avec le plan de projection est nécessaire si on veut faire le lien entre l'interaction 2D et 3D.

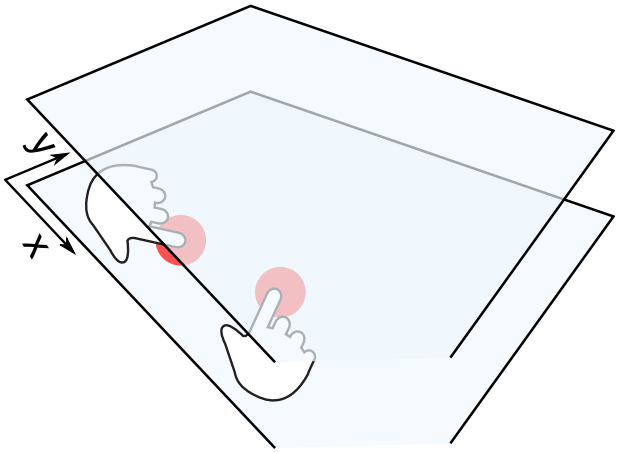


FIGURE 8.8 : Détection du clic dans la couche d'interaction la plus basse

Dans notre système, La détection des clics est activée si on est dans la couche inférieure. Le plan horizontal est défini, et si les extrémités de la main rentrent en contact on émet un clic. Le contact est simplement détecté

si on dépasse une certaine zone verticale.

8.6.3 La reconnaissance des classes de base des gestes

Pour faire le lien avec les études théoriques, notre travail est supposé reconnaître les quatre classes gestuelles de base : toucher, déplacer, saisir, agrandir. Nous sommes arrivés à ce stade à reconnaître la majeur partie des classes de base. Notre système reconnaît pour l'instant la catégorie de geste toucher, déplacer et saisir (partiellement).

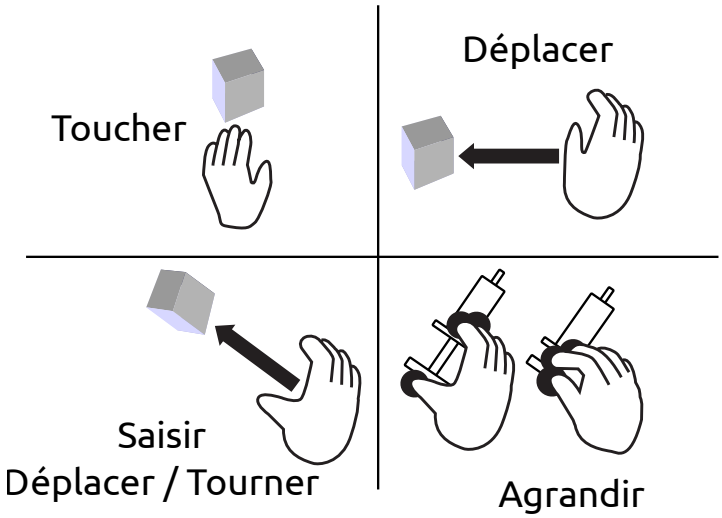


FIGURE 8.9 : Les 4 classes gestuelles à reconnaître

8.6.4 Le suivi de plusieurs mains/utilisateurs

Dans toutes les manipulations qui peuvent se dérouler dans notre espace d'interaction, les utilisateurs se placent à l'extérieur avec leurs mains et bras sont à l'intérieur de la zone de détection.

Bien qu'on arrive à extraire les mains, on peut aussi détecter plus d'informations sur le ou les utilisateurs qui sont entrain de faire les manipulations. On considère que notre système peut être utilisé par 4 personnes

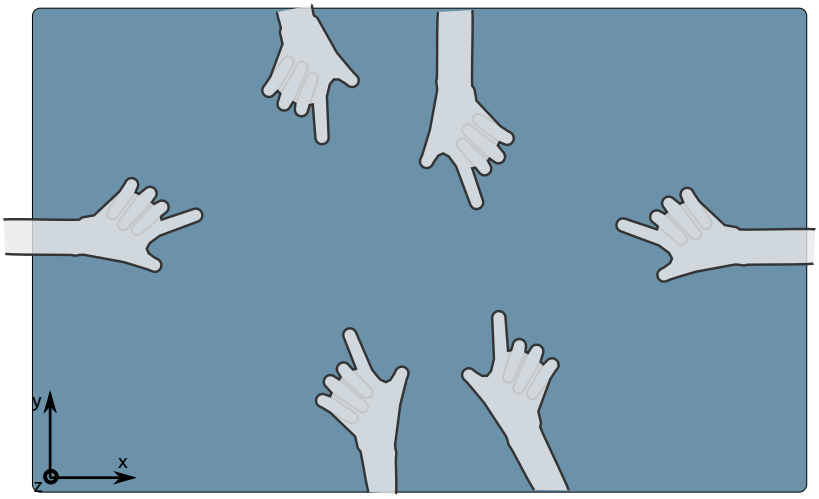


FIGURE 8.10 : Gestion de la reconnaissance multi-main et multi-utilisateur

maximum, et que chaque utilisateur se situe dans une des quatre zones de la table correspondant chacune à un côté de la table. Même si la main est à l'intérieur, son bras guidera toujours au bord où se situe l'utilisateur. Cette détection utilisée est certainement simple, mais elle permet de gérer la majorité des cas possibles dans une interaction multi-utilisateur. On peut comparer cette détection, qui utilise la Kinect, avec la DiamondTouch, une table tactile présente dans le laboratoire. Cette dernière permet de détecter jusqu'à 4 utilisateurs en effectuant un couplage capacitif entre la surface tactile et des récepteurs séparés situés dans la chaise de chaque utilisateur.

Notre système peut remplacer la table, tout en proposant une gestion plus riche au delà du tactile. Au niveau performances, capturer 4 utilisateurs est la limite actuelle puisque les algorithmes ne sont pas parallélisés sur plusieurs processus. La consigne d'utilisation actuelle est d'avoir moins de 6 mains dans la scène, ou d'avoir seulement 2 utilisateurs. Cette limite permet au système de fonctionner sans descendre d'une fréquence de 24 Hz.

8.7 Pointage dans l'espace

Dans une suite du travail et qui peut être complémentaire, on a aussi voulu utiliser le même principe de pointage décrit précédemment mais cette fois sur des objets dans la salle. En généralisant le bloc de pointage sur table, il a été possible de créer un module permettant de pointer sur des objets distants dans toute la salle ATOL qui est le laboratoire basé sur le site de Télécom Bretagne et géré en collaboration avec Thales.

Par soucis de simplification, ce nouveau module propose essentiellement deux nouvelles fonctionnalités :

1. Pointer sur un mur qu'on définit pendant une petite phase de calibrage présentée par la figure 8.11

2. Pointer sur un objet dans la salle une fois qu'on a bien calibré les limites de cette dernière présentée par la figure 8.12.

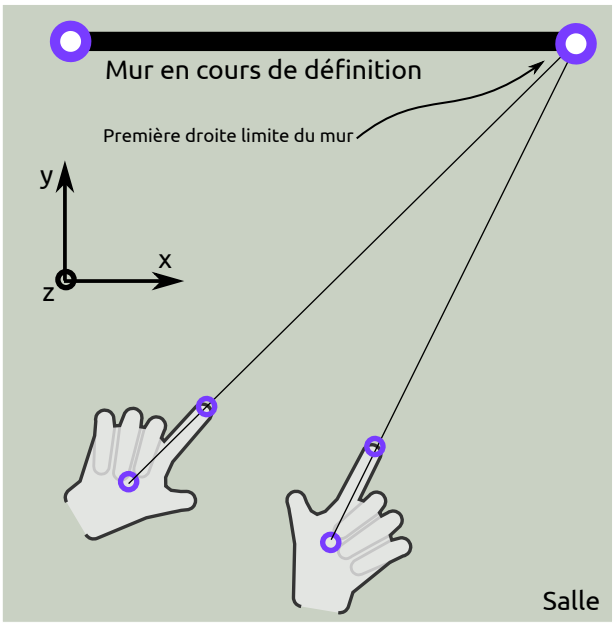


FIGURE 8.11 : Étape de calibrage pour la sélection des limites du mur

Dans le premier cas, on commence par le choix du mur : on pointe avec deux positions de la main vers le premier bord gauche et l'intersection nous donne ses coordonnées dans le repère de la caméra de profondeur. On

commence avec l'hypothèse que les murs sont verticaux, donc l'intersection des droites des mains sont fait en 2D dans le plan (X,Y), l'intersection est un point et le bord est la droite verticale (Z) passant par ce point. Ceci permet d'éliminer la grande complexité de la définition manuelle du plan par un vecteur et dont on entre ses valeurs à la main. Cette méthode élimine aussi la recherche de l'intersection en 3D et la remplace par un algorithme simple d'intersection en 2D. On définit le deuxième bord de la même manière. Une fois le mur défini, on calcule son point d'intersection avec la droite définie par la position en temps réel du centre de la main et l'extrémité du doigt.

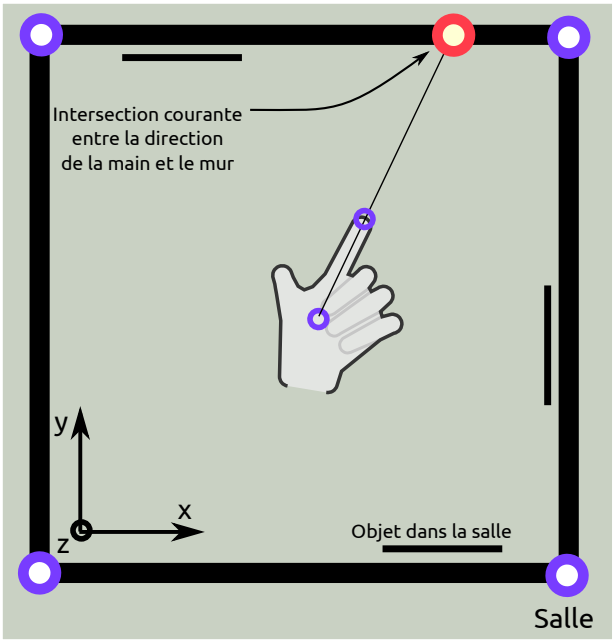


FIGURE 8.12 : La détection de l'intersection entre la main et le mur

Dans le deuxième cas, on calibre les bords des quatre murs de la salle. Une fois les bords définis dans notre système de coordonnées sans à avoir entrer les valeurs numériques à la main, on peut savoir vers quel mur on pointe et sur quel objet si on a déjà défini leurs emplacements.

8.8 Reconnaissance des gestes complets

La reconnaissance d'un cycle de geste complet nécessite un système capable de reconnaître les postures réalisées pendant un déplacement d'une ou deux mains. Dans notre système, la reconnaissance se fait dans un niveau d'abstraction plus bas en émettant les événements des postures et des déplacements en parallèle, et c'est le rôle d'un mécanisme de plus haut niveau de faire la correspondance entre ces informations pour générer une reconnaissance plus complète des gestes.

8.9 Prédictibilité des gestes

Les mouvements qu'un opérateur réalise pour faire un geste n'arrivent pas brusquement d'un seul coup. Ils sont précédés par une étape de réflexion qui décide le geste et par une autre étape de déplacement du bras et de la main pour atteindre la zone du geste.

La forme de la main avant d'atteindre la zone où le geste sera effectué n'est pas neutre. La forme de main en pince pour un geste de saisie commence à être formée par anticipation dans son chemin vers l'objet à saisir. Cette forme de main avant le geste apparent est souvent ignorée complètement par les systèmes de reconnaissance des gestes. L'étude théorique sur le cycle gestuel nous a donné des indications sur les formes de la main avant la saisie, et c'est à partir de cela qu'on a commencé à les intégrer dans notre analyse.

8.10 Évolution avec un autre capteur

Les travaux effectués sont réalisés avec la première version de la Kinect. Elle nous fourni un nuage de points 3D surfacique. La Kinect 2, même si elle a changé de technologie pour rejoindre la famille de capteurs à temps de vol, n'a pas changé la façon avec laquelle elle fournit les données. La sortie de la Kinect 2 est maintenant une surface 3D de 512x424 points.

Cette surface qui semble inférieure à celle de la Kinect 1 de 640x480, est plus précise et avec moins de perturbation. En terme de temps de calcul, dans tous les cas, on devrait traverser la surface point à points pour faire le marquage et ensuite traverser les mains captées. Le temps de calcul avec la Kinect 2 est supposé être plus rapide et plus efficace.

8.11 Conclusion

Dans ce chapitre, on a expliqué quels sont les critères les plus importants à reconnaître dans les postures et mouvements de la main. On a expliqué la signification de la reconnaissance d'une posture et d'un geste, et détaillé la procédure technique pour le réaliser à partir de notre matériel. L'aspect de la prédictibilité d'un geste avant sa réalisation est un détail important mais malheureusement pas encore très développé dans notre travail.

Chapitre 9

La Manipulation 3D dans la pratique

9.1 Introduction

Dans ce chapitre nous allons présenter deux applications développées pour tester nos algorithmes gestuels. Ces applications vont consommer les événements transmis du système de reconnaissance des gestes et vont les utiliser pour activer certaines fonctionnalités. La première application utilise les algorithmes dans le cadre d'une surveillance maritime avec une interface bidimensionnelle. La deuxième application permet de manipuler des objets en 3D. Ce chapitre présente les difficultés rencontrées dans un cas pratique. Notre travail touche toute la chaîne gestuelle et dans ce cas c'est dans la liaison entre un système gestuel et des applications. La correspondance n'est pas toujours directe entre le message gestuel et l'action voulue.

9.2 Application des gestes 3D dans un contexte 2D

9.2.1 Architecture et choix techniques de l'application

Pour cette application, on a choisi d'utiliser la bibliothèque Qt pour créer l'interface et communiquer avec d'autres modules nécessaires pour générer les éléments graphiques.

On a choisi RECONSURVE, un projet de recherche mené au laboratoire qui concerne la surveillance maritime, comme application cible qui va consommer les informations gestuelles transmises par notre moteur de reconnaissance.

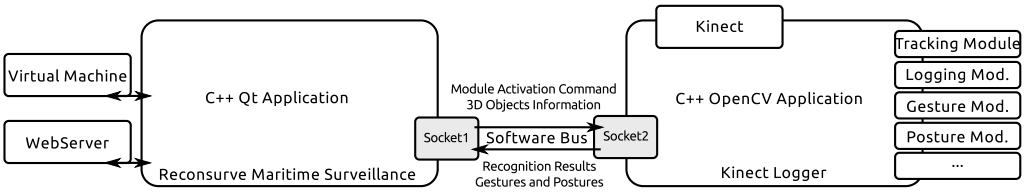


FIGURE 9.1 : Architecture de l'application 2D consommant les gestes

Le projet RECONSURVE dans son prototype initial utilise une carte d'une zone maritime qui contient les positions des bateaux et les affiche en communicant avec une base de donnée. L'application reçoit les positions des bateaux en envoyant des requêtes à la base de donnée et en recevant les informations sous forme d'un flux XML. Ce flux contient les positions GPS et fait les conversions nécessaires de la position GPS vers les coordonnées sur l'écran. L'interface 2D a été développé en utilisant la bibliothèque Qt et plus précisément la "QGraphicsScene". L'application accède à la carte depuis un service Web et fait le rendu en jouant le rôle d'un navigateur Web. Le composant web ajouté en arrière plan de la scène supprime les éléments non nécessaires qui sont dans l'interface originale et contenant les logos du site via l'envoi de commandes Javascript au composant. Les bateaux

et les autres éléments sont ensuite ajoutés comme des composants à part dans la scène au dessus de cette carte. On fait aussi une interception des commandes d'entrée clavier souris puis on les émet deux fois, une fois vers le composant web de la carte, et une deuxième fois sur la scène pour avoir une interactivité comme si tous les composants de l'application appartenaient à la même unité.

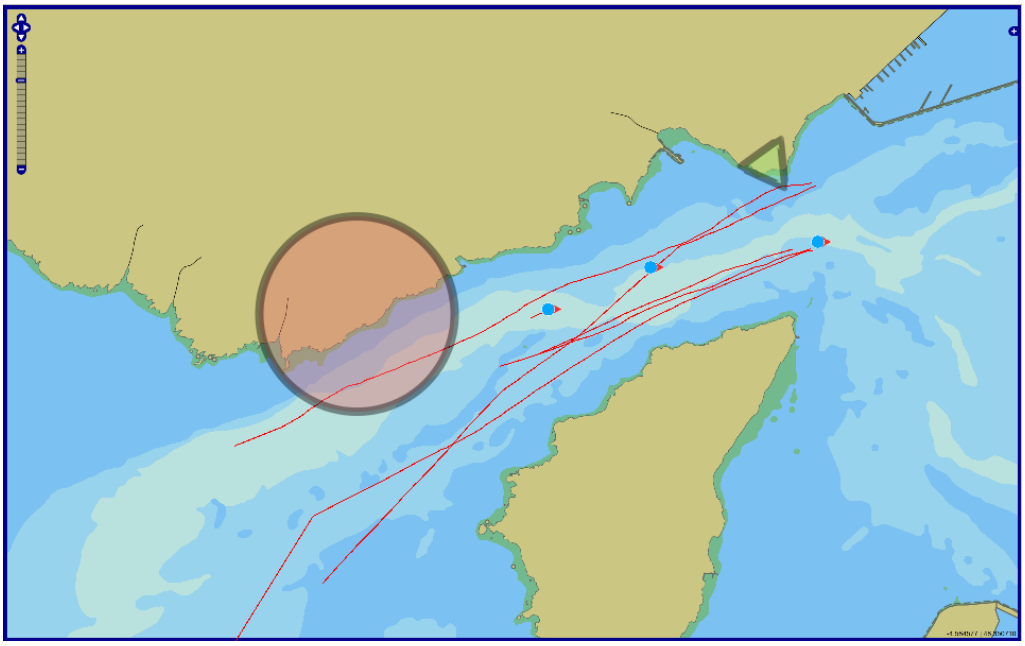


FIGURE 9.2 : L'interface de l'application 2D développée

L'interface finale contient des zones interdites où les bateaux ne doivent pas entrer. Les bateaux ont une trajectoire qu'il faut afficher sur la carte et animer. Les bateaux qui présentent une trajectoire non habituelle, avec un motif d'accélération décélération ou avec des angles aigus sont mis à jour dans l'interface pour afficher une alerte et informer sur leurs positions actuelles. L'application consomme les événements gestuels en activant un mode de commande et en émulant des raccourcis clavier/souris.

9.2.2 Scénario de test

Le scénario de test minimal développé consiste à ce que des utilisateurs commandent le système avec leurs gestes au dessus de l'interface. Les utilisateurs génèrent des tracés 3D dans l'espace pour par exemple faire clignoter un bateau ou une zone.

La liaison de la commande gestuelle utilisant des tracés dans l'espace avec l'interface 2D était faite rapidement pour évaluer d'abord la possibilité de lier deux environnements ensemble. Le pointage sur un élément sur la table a été utilisé pour commander un drone d'aller à un emplacement donné. Seulement les déplacements de la main dans l'espace et leurs reconnaissance ont été utilisés pour activer un algorithme et faire clignoter un bateau ou une zone interdite selon le geste. L'utilisation des tracés dans l'espace n'a pas été fructueuse. Ils sont longs à produire, difficiles à apprendre et à retenir et sans correspondance claire dans le modèle mental des utilisateurs. Les premiers tests nous ont poussés à ne pas continuer dans cette direction et ne pas considérer ce cas d'utilisation.

9.3 Application des gestes 3D dans un contexte 3D

Dans une deuxième application créée pour consommer les informations gestuelles, on a pensé à utiliser des objets 3D dans une scène et utiliser la Kinect à distance pour les déplacer et les saisir.

L'application devrait afficher une carte côtière en 3D avec des zones sphériques et des tubes ainsi que les bateaux suivant des trajectoires dans le temps. Les zones sphériques représentent des zones protégées, et les tubes représentent des couloirs pour les drones. Dans ce prototype de test, on a utilisé principalement la reconnaissance du geste de la saisie pour l'utiliser dans le déplacements des objets.

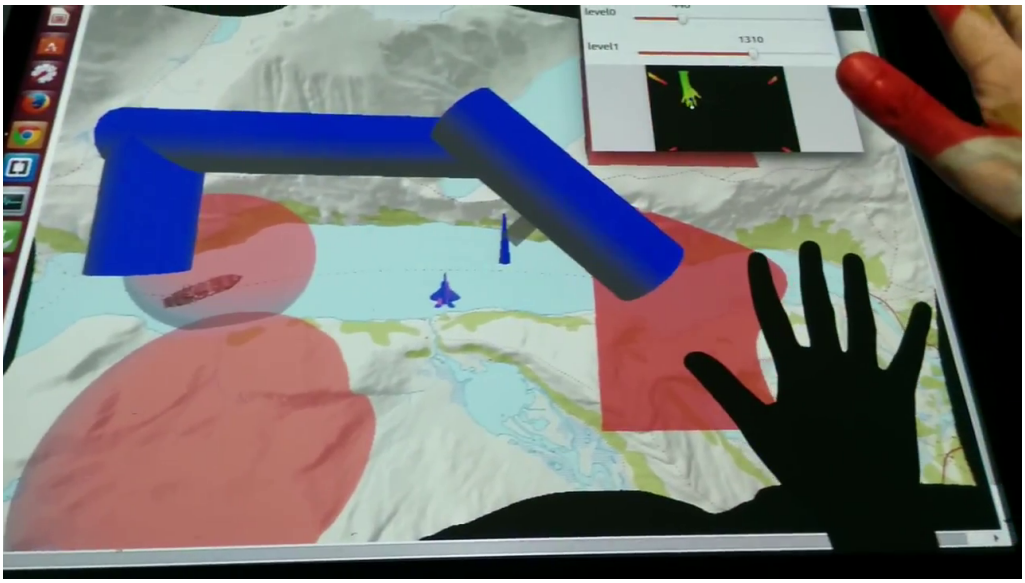


FIGURE 9.3 : La projection de l'interface 3D sur la table

9.3.1 Architecture de l'application

Pour cette application nécessitant la 3D, on a parcouru les options technologiques disponibles pour arriver à un résultat rapide à réaliser et offrant une bonne qualité. Après avoir testé plusieurs moteurs 3D, et après avoir testé la 3D avec Qt, on a découvert qu'il fallait apprendre des notions bas niveau en 3D comme les "Shaders" [1] pour pouvoir créer une scène simple. On a donc continué à chercher d'autres solutions. À la fin on a eu le choix entre utiliser soit Unity qui est un moteur 3D simplifiant la création des scènes avec ses interfaces soit WebGL combiné avec la bibliothèque Javascript Three.js. Notre système fonctionne sous Linux, et avec Unity il n'est pas possible d'éditer un projet sous ce système. On s'est donc tourné vers l'utilisation de WebGL dans le navigateur combiné avec Javascript.

La communication entre le système gestuel et la fenêtre contenant les éléments 3D dans le navigateur est effectuée grâce à un lien en WebSocket. La bibliothèque libwebsock a été intégrée dans le système gestuel qui

1. Les shaders sont un code similaire au C, et qui sera compilé et exécuté sur les blocs de traitement la carte graphique

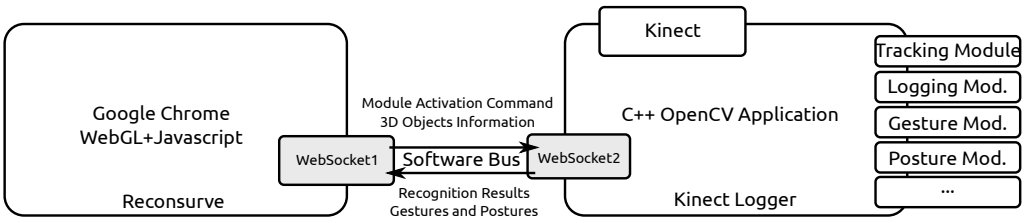


FIGURE 9.4 : Architecture de l'application 3D consommant les gestes

émet ses informations dans une structure JSON. Cette dernière contient les positions du centre de la main en 3D et le geste reconnu.

9.3.2 Scénario de test

Le scénario de test minimal pour cette application était lui aussi lié au projet RECONSURVE et la surveillance maritime. Plus de blocs gestuels ont été liés, et l'utilisateur pouvait déplacer des zones demi-sphériques au dessus de la carte ainsi que déplacer des cylindres qui représentent des couloirs aériens.

Le continuum 2D-3D a été exploité en partie, puisque l'utilisateur manipule les cylindres dans l'espace, mais les demi-sphères sur le plan. Le déplacement d'un cylindre nécessite l'utilisation de la posture de saisie de la main, mais le déplacement des zones nécessite une posture plutôt plane de la main et un touché du plan. Un objet n'est déplacé ou modifié que s'il est proche de la main. Cette dernière est représentée par une ligne montrant sa direction principale.

Ce scénario illustre l'état auquel on est arrivé dans notre traitement de l'entrée Kinect depuis la réception des nuages 3D brutes, la segmentation de la main, la reconnaissance de la posture et l'envoi des événements vers une autre application en temps-réel à travers la WebSocket. À ce stade, il reste encore des voies d'amélioration dans la stabilité de la reconnaissance puisqu'une erreur dans les couches antérieures peut être amplifiée par la suite. L'ajout de modules probabilistes pour filtrer la reconnaissance peut réduire les erreurs dans la reconnaissance.

9.4 Critiques et visions d'améliorations

9.4.1 La notion de couches d'interactions

La notion des couches d'interactions est un concept qui permet d'avoir une différentiation dans l'interprétation des gestes en fonction de la proximité de la main envers cet objet.

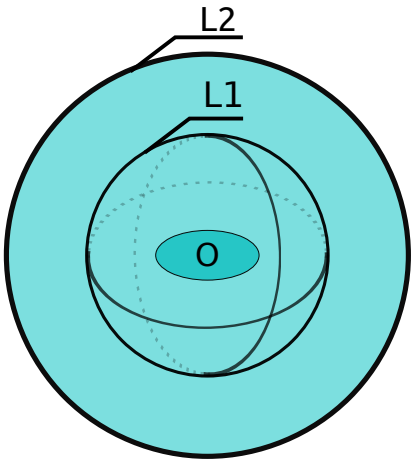


FIGURE 9.5 : Les couches d'interaction autour d'un objet

On a proposé la notion de la sphère d'interaction comme montré dans la figure 9.5 pour pouvoir appliquer la manipulation directe sur l'objet. Cette manipulation directe ne s'active que si la main est assez proche de l'objet. On définit la première couche de la sphère autour de l'objet pour l'interaction directe. Dans cette zone, l'opérateur peut déplacer, agrandir, tourner et sélectionner l'objet. Dans la deuxième couche d'interaction, l'opérateur peut utiliser une posture comme commande. Quand la main n'est pas dans les zones autour de l'objet, on peut ignorer sa configuration.

On voulait faciliter les interactions directes avec les objets dans le sens des manipulations. Dans un cas pratique, la connaissance de la position de la main et de l'objet peuvent simplifier la reconnaissance et l'implémentation des algorithmes. Par exemple, pour la rotation, au lieu de suivre les changement sur la main, on suit seulement la densité du nuage du point de

la main par rapport au centre de l'objet pour détecter la rotation, ce qui est rapide à calculer.

9.4.2 Le déclenchement des modules de reconnaissance

Après traitement des données bas niveau et la construction des algorithmes de reconnaissance sous forme de modules, on est en face du problème du lancement mutuel des modules. Dans un cas particulier, on peut avoir plusieurs modules de reconnaissance qui sont lancés en même temps et qui donnent des résultats différents. Une bonne solution à ce problème est de créer un composant qui décide qui sera le module à exécuter ou si les modules sont lancés ensemble, quels messages sont à envoyer et quels sont ceux à ne pas propager selon le contexte.

Dans le cadre de la couche applicative uTouch d'ubuntu, on a eu ce problème mais la décision est de ne pas le traiter et de pousser son traitement vers les applications. Une solution était d'envoyer tous les événements gestuels reconnus, et c'est à l'application de s'abonner aux gestes voulus, et ignorer les autres. L'application peut aussi s'abonner à tous les événements et avoir un module interne qui décide quel geste utiliser selon les objets dans leurs contexte et leurs positions. Une bonne résolution d'un tel problème ne peut pas exister sans l'accès aux composants internes de l'application et avoir un système de choix. Il est encore un sujet de recherche à explorer.

9.4.3 Le prototype représenté en couches applicatives

Notre système avait à la base l'objectif de gérer tout le cycle d'interaction. L'objectif est assez ambitieux et sa mise en pratique peut démontrer des problèmes.

D'après la figure précédente **??**, on voit qu'il y a une partie contenant plusieurs couches. Les résultats d'une couche supérieure dépend de celle

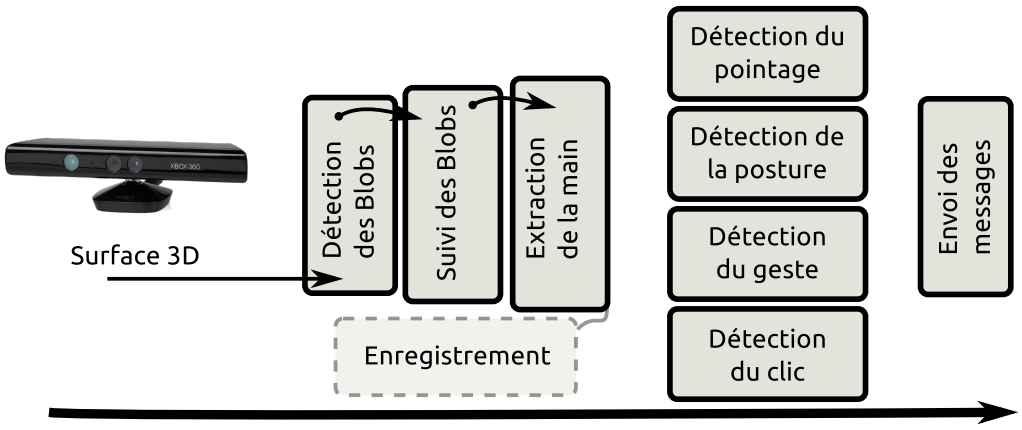


FIGURE 9.6 : Le système de reconnaissance représenté en hiérarchie de modules

qui la précède. Ceci signifie aussi que les petits problèmes dans une étape peuvent être amplifiés au fur et à mesure qu'on traverse les couches.

9.4.4 Occultation par la main

Le problème de l'occultation par la main est l'un des problèmes les plus visibles que nous avons rencontrés avec notre installation. Dans notre cas, le projecteur projette de sa position supérieure vers la table. Lorsque la main rentre en interaction, elle occulte ce qui se passe en bas. Dans le cadre d'une interaction sur table, où la main n'est pas trop élevée, ce problème est minime, mais plus elle s'élève en 3D, plus elle cache une grande partie de la scène.

La figure 9.7 montre comment plus on élève la main, plus on a le problème de l'occlusion sur la table. La Kinect est aussi affectée par ce problème, la taille des zones de la main est plus grande puisque la Kinect elle aussi fonctionne en projetant un nuage de point dans son mécanisme de capture. Nous étions limités dans notre travail par cette installation parce qu'il y avait un objectif initial d'utiliser des projecteurs 3D stéréoscopiques actives. Dans ce cas précis, nous avons aussi un problème similaire d'interaction même en utilisant des écrans LCD car la main interagit de façon

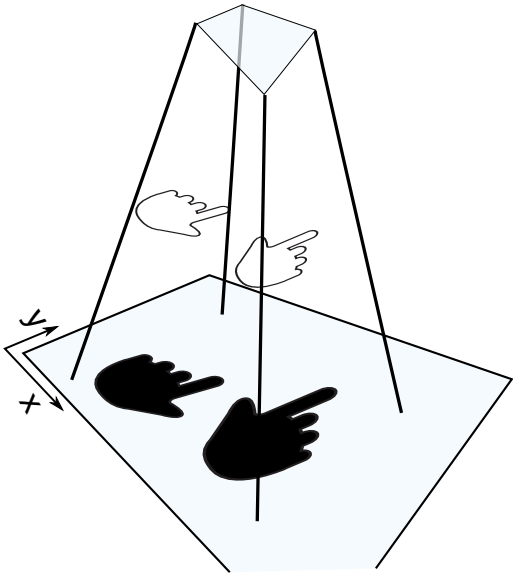


FIGURE 9.7 : L'occlusion par la position de la main

décalée.

9.5 Conclusion

Dans ce chapitre, on a fait une liaison entre un système qui fournit des messages d'entrées gestuelles, et des applications qui consomment ces événements. On a expliqué les architectures de chacun, ainsi que la façon avec laquelle on a assuré la distribution des messages. La liaison n'a pas été une correspondance directe, et les performances après avoir testé sur un cas pratique sont à discuter. L'utilisation des gestes tridimensionnels avec simplement une interface bidimensionnel projetée n'a pas été performant en comparant avec des moyens traditionnels comme le clavier et la souris. Et dans le cas 3D, il est important que l'application ait une communication dans les deux sens avec le système de reconnaissance. Cette communication permettra d'améliorer les algorithmes de détection en ayant une notion sur les objets 3D.

Chapitre 10

Conclusion

Au cours de cette thèse nous avons eu comme objectif de mener un travail pour la reconnaissance des gestes. Pour ceci, notre première contribution est d'organiser les travaux précédents impliquant de prés ou de loin une relation avec le geste. Nous avons étudié le cycle de sa production en commençant par l'humain. Nous avons essayé d'étendre le domaine d'étude gestuel et de proposer une nouvelle façon d'assembler les différents travaux liés à la production de gestes. Nous avons mis en perspective les résultats issus de plusieurs domaines de recherche. Dans l'état de l'art on a aussi utilisé le cycle de Norman comme support pour placer les différentes études. Nous avons aussi été poussé à commencer par l'application de ces études en prédisant les gestes de la main avant de toucher l'objet. Nous avons fait un recensement des périphériques, algorithmes et méthodes pour la détection des mouvement humaines et les transformer en des informations utiles.

Nous avons également développé un système mettant en œuvre ce cycle de geste étendu et pouvant être utilisé pour réaliser des analyses de gestes. Notre système arrive à transformer une entrée de bas niveau à des informations de mouvements et de gestes. Ces informations ont été utilisées dans un cas pratique dans un environnement 3D. Ce travail sur le cycle complet nous a délivré un retour d'expérience des réussites, échecs et méthodes encore à améliorer pour maîtriser l'interaction.

Dans un chapitre placé en annexe vu son caractère “génie logiciel”, nous avons présenté une vision de la façon avec laquelle on peut améliorer la création d'application évolutive. Ce dernier chapitre reste important dans l'étude pour clôturer le cycle du geste avec la mort et la consommation de ce geste et sa transformation en une action sur un objet.

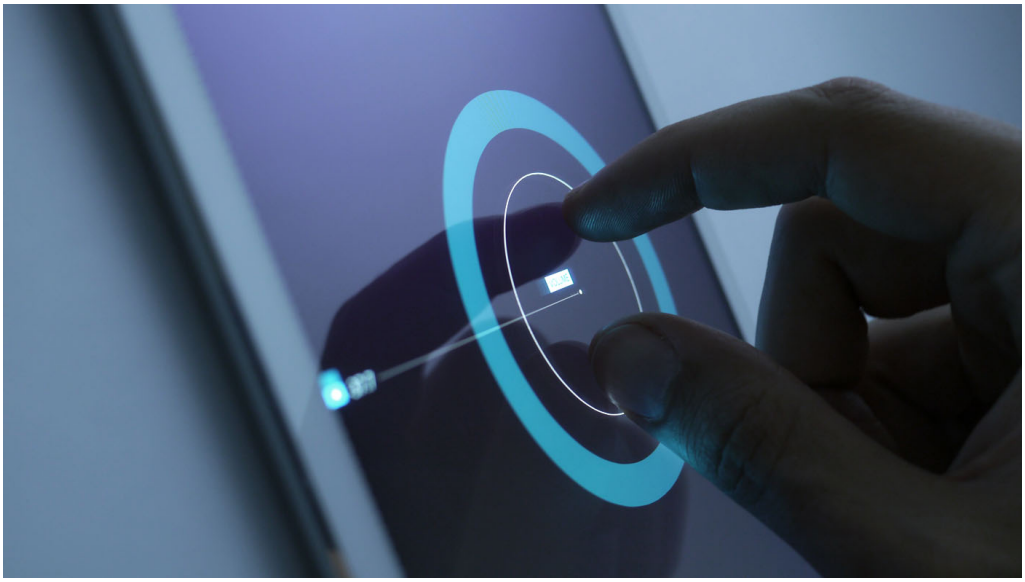


FIGURE 10.1 : L'utilisation des gestes dans la voiture sans nécessiter de voir l'interface

Bien que l'étude menée concerne les gestes pensés après la vision d'un objet, il faut aussi dire qu'il y a des cas où le contexte oblige de pas regarder l'interface. Dans un concept présenté par Matthaeus Krenn [1] et présenté dans la figure 10.1, on a une interface gestuelle de commande de voiture où il enlève la nécessité de voir l'écran. L'accès aux commandes se fait avec le nombre de doigts sur écran. Chaque configuration active un mode de commande soit pour augmenter ou réduire le volume, modifier la source ou changer la température. L'utilisateur modifie chaque option sans regarder l'écran en bougeant ses doigts en haut et bas, augmentant l'ouverture des doigts ou en faisant une rotation.

1. http ://matthaeuskrenn.com/new-car-ui/

Ce genre d'applications pourra également bénéficier de nos propositions en s'inspirant de la saisie humaine pour enrichir les types de commandes et d'éliminer la nécessité de les retenir en ayant une métaphore avec l'utilisation d'un objet réel comme a fait Chris Harrison [Harrison et al., 2014].

Il est aussi à noter que beaucoup de problèmes restent ouverts surtout ceux qui sont liés à l'ingénierie des applications interactives. Notre travail montre qu'il ne suffit pas de créer un simple moteur de reconnaissance des mouvements finaux visibles, mais de suivre la création antérieure et la consommation postérieure à l'intérieur des applications. On a cité le problème du déclenchement de façon concurrente des événements gestuels et de pousser le filtrage dans les applications ou dans un autre bloc applicatif. Ce dernier peut recevoir ses messages et être capable de reconnaître des expressions gestuelles de plus hauts niveau, appelé dans le contexte du développement ubuntu "gesture continuation". Enfin, il est aussi à noter que beaucoup de problèmes restent ouverts sur tout ceux qui sont liés à l'ingénierie des applications interactives.

Annexe A

Au delà du geste, les applications interactives

A.1 Introduction

Le sujet de la reconnaissance des gestes et de l'estimation des intentions d'interaction d'un utilisateur avec les objets est un sujet assez large. Par contre, la durée de vie d'une interaction gestuelle qui commence dans le cerveau de l'utilisateur ne se termine après la reconnaissance de ces intentions. Un geste sera ensuite consommé par une application et doit le traiter et délivrer un retour.

Dans la petite expérience qu'on a eue avec la manipulation gestuelle, on a pu avoir un aperçu sur une autre façon de créer des applications. Bien que cette expérience se limite pour l'instant dans le système d'exploitation Linux, les grandes lignes de l'idée peuvent être transférés ailleurs et aider à mieux architecturer une application pour qu'elle soit plus interactive.

Comme la majorité des développeurs, j'ai appris à utiliser et développer pour les systèmes UNIX tel que Linux. Parmi les principes de ce système est que chaque application en ligne de commande est conçue pour accomplir une seule tâche, l'accomplir bien, et avec la possibilité de communiquer avec les autres petites applications. Grâce à ce principe, il a été possible

d'accomplir des tâches complexes avec la combinaison de ces petits outils. Malheureusement, cette efficacité n'est possible qu'avec les applications en ligne de commande sans interface graphique, les applications graphiques sont des îles séparés quasiment sans inter-communication.

Dans ce chapitre, on propose une nouvelle solution pour rendre ces applications graphiques plus communicantes, plus efficaces, facilement scriptables, et plus adaptables aux nouveaux périphériques gestuels. On propose cette solution à travers des études de cas mais aussi des scénarios implémentés en preuve-de-concept. Notre solution repose sur l'utilisation d'un bus logiciel pour orchestrer les communications inter-applications. Dans nos exemples, on vise les designers dans leurs applications créatives comme un point de départ pour une réflexion plus large. Les scénarios sont ordonnés par leurs complexité.

A.2 Applications en tant que canevas

A.2.1 Définition d'un bus logiciel

Un bus logiciel est un système de communication inter-processus qui permet à plusieurs différentes applications de se connecter à un canal commun et ainsi passer des information ou invoquer des appels de méthodes distantes. Dans le cas des systèmes Linux, D-Bus est utilisé pour faciliter la communication de trois types de messages : appels de méthodes, signaux et propriétés. Il est possible d'appeler une méthode dans une application distante, s'inscrire à ses signaux sous forme d'événements émis, ou lire les variables de propriétés partagés sur le bus. Comme défini par le standard, chaque application a une adresse et une liste de méthodes qui peuvent être examinées par n'importe quel autre application qui lit sur le bus. Le choix de D-Bus est suggéré par le nombre d'outils, bibliothèques et applications qui le supportent déjà.

A.2.2 Anciens travaux liés

Ayant participé aux travaux de développement autour du support de l'émission des événements multi-tactiles sous Linux, j'ai été confronté au problème de modifier une large liste de bibliothèques et de couches logicielles pour supporter le routage des événements entre le noyau et l'application finale. J'ai opté pour une solution plus rapide que j'ai développée et appelée "Ginn". C'est un daemon Linux qui transforme les événements multi-tactiles vers des clics clavier et souris. La transformation suit des règles écrites dans un fichier de configuration en XML appelé "wishes". Une configuration est chargée selon l'application actuellement au premier plan. Le concept de la transformation des événements d'entrée s'inspire du travail de recherche appelé "iCon" par Pierre Dragicevic [Dragicevic and Fekete, 2004].

Ginn, qui est en gros un petit morceau de code a permis aux anciennes applications d'avoir une apparence qu'ils réagissent aux nouveaux événements, et sans modifier leurs propres code ou demander aux développeurs respectives de le faire. Ginn a proposé une solution pour adapter les anciennes applications vers des nouvelles façons d'interaction, mais il a besoin d'avoir l'application affichée au premier plan, et il a besoin aussi de leurs anciens interfaces, là où les événements seront injectés. De ce cas, j'ai senti l'importance d'inclure des bouts de codes permettant aux applications d'évoluer dans le futur pour supporter de nouveaux moyens d'interaction comme expliqué par Stéphane Chatty [Chatty et al., 2007]. Des moyens autres que les événements cliver et souris. Ginn aurait pu avoir plus de liberté s'il était capable de déclencher, à travers une certaine règle, une fonction interne dans l'application cible. Dans ce cas précis, il a pu contourner l'interface proposée et n'aurait plus besoin que du "canvas", coeur de l'application. Cette architecture est présenté dans la figure A.1.

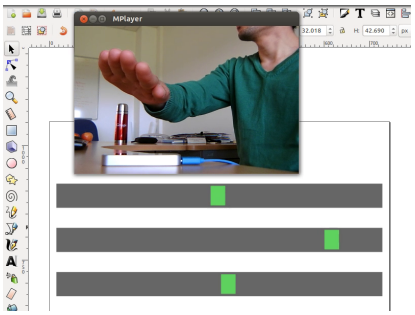


FIGURE A.1 : Connection des événements de rotation de la main pour contrôler les mouvements des carrés dans inkscape, aucune modification du code de l'application n'est fait.

A.2.3 Réduire les applications à leurs canevas

Les applications graphiques ont éliminé le besoin de mémoriser les commandes pour accomplir les tâches. En même temps, ils ont aussi éliminé la possibilité de les scripter ou de les sérialiser d'une façon simple comme dans la ligne de commande. Quelques tentatives ont essayé d'utiliser les bus logiciel comme TUIO [Kaltenbrunner et al., 2005] ou IvyBus [Buisson et al., 2002] pour ajouter une couche de communication. Dans le premier cas, pour transporter les événements de l'entrée qui sont plus complexes que ceux standardisé dans le système d'exploitation. Et dans le deuxième cas pour transporter des informations de types génériques. Olivo et al. [Olivo et al., 2011] ont parlé du problème général quand ils ont travaillé sur la gestion des événements multi-tactiles, et ils ont utilisé les protocoles réseaux. Ceci ne va pas résoudre le problème, surtout si plusieurs applications n'utilisent pas les mêmes standards de la plate-forme.

Dans ce travail, on présente comment réduire l'autorité de l'interface des applications sur ce que l'utilisateur a le droit de faire. Notre approche est différente de celle de Stuerzlinger et al. [Stuerzlinger et al., 2006] qui ont essayé d'innover dans le niveau interface pour des fins similaires. Minimiser le rôle de la majorité des applications vers un canevas de visualisation est réalisé en accédant aux méthodes internes depuis un bus logiciel. Ce principe permet de simplifier la migration de l'application vers un nouveau

paradigme post-wimp, car on va avoir besoin de connecter les nouveaux événements d'entrée (comme les gestuels) vers ces méthodes internes, et on va avoir le canevas que pour le retour utilisateur. Contrairement au patron de design Modèle-Vue-Contrôleur, où les frontières d'une seule application sont déjà connues à l'avance, on est en train d'adresser une plate-forme avec multiple applications et qui peuvent être utilisés sur différents périphériques. Ceci n'est pas forcément comment les développeurs originaux ont prévu l'utilisation.

Dans la majorité des scénarios suivant, on vise principalement l'application open-source de dessin vectoriel Inkscape, et on montre les nouvelles possibilités qui en découle.

A.3 Scénarios

A.3.1 Scripter des applications graphiques

Les applications graphiques ne peuvent être utilisées qu'à travers leurs interfaces, et en utilisant les périphériques d'entrée qu'elles supportent. Lorsque nous voulons gérer une manipulation complexe, nous sommes coincés dans la répétition des actions d'entrée, en particulier lorsque les développeurs n'ont pas fourni ni un enregistreur de macros ni une API de script interne comme VBA ou Gimp PDB (Procedural DataBase). Les solutions à ce problème est d'utiliser un démon externe pour enregistrer et répéter les clics clavier/souris, ou d'autres plus complexes comme celui du MIT Sikuli [Yeh et al., 2009] qui utilise la vision par ordinateur pour détecter la position des éléments de l'interface graphique, puis génère des clics sur ces éléments. Utiliser la vision par ordinateur pour une telle opération signifie que nous avons encore besoin de l'interface graphique affichée en premier plan à l'écran. Cela ressemble plus à un hack, qu'à une solution permanente, puique l'interface peut changer à tout moment entre les versions. Dans notre cas, en accédant aux méthodes de l'application, exposées sur un bus logiciel standardisé, nous sommes en mesure de scripter toutes les

actions posibles en utilisant un simple script Python exécuté à l'extérieur de l'application elle-même. Cette application est simplement réduite à un canevas de visualisation des effets de ce script.

A.3.2 Programmation interactive

Quand on scripte une application comme on a discuté précédemment, on a la possibilité de combiner plusieurs de ces bouts de code, les rendre configurable et créer un interface graphique pour eux. En prenant l'exemple d'une application de dessin comme Inkscape, cette dernière est constituée d'un canevas là où les éléments graphiques sont dessinés, et un interface utilisateur fourni par défaut. On peut utiliser des fenêtres flottantes de la même façon que les "Magic Lenses" par Bier et al. [Bier et al., 1993] pour ajouter de nouvelles fonctionnalités. Ces fenêtres flottantes vont contenir un script pour gérer les objets contenus dans Inkscape comme dans la figure A.3. Ces fenêtres peuvent utiliser des curseurs pour configurer les valeurs d'une forme géométrique complexe. Les commandes DBus sont ensuite générées, sérialisées pour être envoyées sur le bus envers l'application. Avec les méthodes simples de l'application disponibles sur le bus comme "dessiner ligne" ou "dessiner cercle", on peut maintenant développer des plug-in externes permettant de dessiner des formes complexes. La communication se fera à travers le bus, et ces plug-ins peuvent être créés dans n'importe quel langage tant qu'il émettent les messages sous la forme du standard.

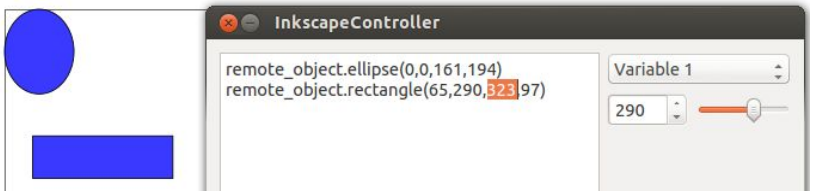


FIGURE A.2 : Implémentation du dessin interactive de figures géométriques. L'utilisateur peut sélectionner une valeur et visualiser le changement de façon instantané dans Inkscape pendant qu'il bouge le curseur.

Cette nouvelle façon d'ajouter des fonctionnalités aux applications est

très générique et permet à l'application de se comporter d'une nouvelle façon. Par exemple, une application de dessin vectoriel peut gagner la possibilité de se comporter comme une application de conception ou de dessins d'architecture. Un autre exemple permet de transformer une application de dessin simple vers une application de traçage en lisant les valeurs depuis un fichier et les transformer vers des commandes de dessin. Nous avons créé une visualisation animée en utilisant la méthode "déplacer" sur un objet graphique. Le plug-in extérieur exporte à chaque fois une image après le déplacement et à la fin combine les images pour créer une vidéo. D'un coup, l'application Inkscape a acquis la possibilité de générer des vidéos d'animation en branchant un plug-in extérieur et grâce à la composition des méthodes. La figure A.2 montre l'utilisation des langages C++ ou de Python pour ajouter l'interactivité.

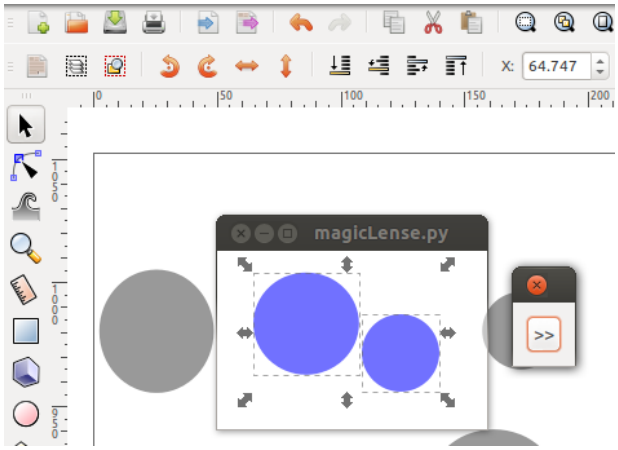


FIGURE A.3 : Implémentation du concept des "magic lenses". L'application complètement séparés a une fenêtre semi-transparente et peut modifier la couleur des éléments qui se situent derrière son cadre dans Inkscape

A.3.3 Factorisation du développement d'applications

Dans l'exemple précédent, on a atteint un niveau où on peut amener une application graphique à se comporter d'une nouvelle façon en utilisant seulement des plug-ins externes qui ne dépendent pas sur l'interface fourni

par défaut. Ceci signifie que la partie la plus importante d'une application a changé. Quels seront les limites si on pousse ce concept envers un nouveau niveau par la suppression complète de cette interface par défaut et de fournir des nouveaux ?

Tout concepteur utilise de nombreux outils pour créer une maquette ou une animation. Les outils qu'il utilise ne viennent pas toujours d'une seule entreprise. Ainsi, il a besoin d'apprendre à utiliser l'interface de chacun d'eux. Un débutant trouvera que ces outils ne partagent pas les mêmes icônes, le même design d'interaction, même le même positionnement des éléments de la barre d'outils qui est un problème d'incohérence dans la plate-forme de l'artiste. Ce problème peut être résolu lorsque le développeur d'une plate-forme peut accéder à des fonctions internes à travers les logiciels installés et fournir par lui-même une interface qui contient le même ensemble d'icônes et basé sur une même conception de l'interaction. Et un tel accès peut être fait en utilisant un bus logiciel entre les applications.

Nous sommes en train de proposer une résolution de l'inconsistance dans une plate-forme. Mais de nos jours, les applications peuvent être exécutées sur les téléphones mobiles, les tablettes, les ordinateurs de bureau et les télévisions. Même si le type d'application peut différer d'un dispositif vers un autre, on a encore un espace commun de fonctionnalités utilisées. Pour pouvoir cibler tous ces dispositifs, un développeur créerait une application pour chacun, avec une interface et une interaction propre, les compiler et fournir une nouvelle version. Le problème ici est qu'il y a besoin de beaucoup de travail et de compétences différentes du développement à la conception d'interaction. Lorsque celui qui fait le portage d'une application vers un autre dispositif est différent, il devient parfois difficile de convaincre le développeur principal de créer une nouvelle interface. Ce scénario applique les deux niveaux de séparation : Les fonctionnalités de base incluant un canevas de visualisation, et une interface superposée. La base où le "core" exporte les fonctionnalités vers le bus logiciel, et l'interface se connecte à ce bus, charge une interface spécifique ainsi qu'un “processus de correspondance

de fonctionnalités" suivant le dispositif. Ce concept est représenté par la figure A.4.

Les interfaces sont créés par les développeurs principaux de la plate-forme logicielle, et la phase d'intellection va se promouvoir du niveau application vers le niveau plate-forme. Le travail requis pour adapter l'application vers un nouveau dispositif ou modalité est réduit à la création de l'interface et à la correspondance. Le rôle du cœur de l'application sera de montrer un canevas, et d'exporter les méthodes internes vers le bus. Donc dans ce paragraphe, à part la résolution de l'inconsistance dans une seule ou multi plate-forme, nous avons factorisé le développement des applications dans le cœur et factorisé aussi le design d'interaction qui sera fait suivant les mêmes règles logiques du designer sur toutes les plate-formes. Le lien entre les deux couches est un composant qui va faire la correspondance des interactions avec la fonctionnalité concordante. En pensant de la plate-forme et pas de l'application en soi, on peut pousser le concept à la visualisation du même canevas de l'application vers plusieurs dispositifs connectés, tout en ayant la possibilité de modifier les éléments en même temps à travers l'ordinateur, la tablette et le téléphone. L'utilisateur pourra faire une partie du travail dans un dispositif et la finir en utilisant un autre. Les événements seront traités dans chaque périphérique et transformés vers des appels de méthodes DBus.

A.3.4 Compositeur d'applications

Dans une plate-forme où plusieurs applications auraient leurs méthodes internes exportées sur le bus et gérées par les mêmes conventions de design d'interactions, il y aura besoin d'un outil capable de se connecter à n'importe quel méthode d'une application. Ensuite, faire fonctionner des tâches plus complexes que celles fournies par défaut par l'application. Cet outil capable par exemple d'ouvrir un ficher utilisant la méthode "ouvrir fichier" exportée, applique des opérations dessus et enregistre les résultats dans un nouveau fichier. Ensuite après avoir fini avec la première application, ouvrir

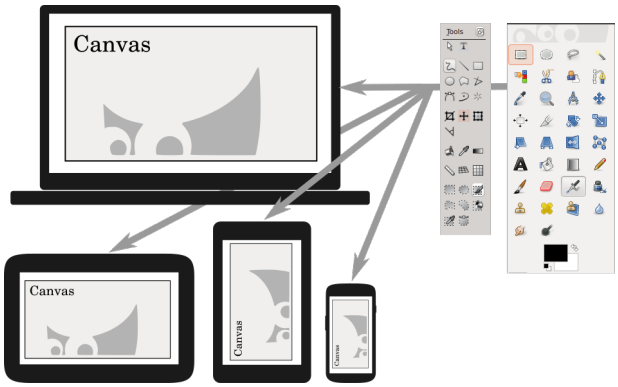


FIGURE A.4 : Concept du développement d'une application multi-périphérique où l'application fournit un canevas de dessin et exporte ses fonctionnalités dans le bus

une nouvelle et ainsi de suite. Cet outil permettra de rendre les applications graphiques aussi puissante que les commandes UNIX, voire même avec plus de possibilités. Puisque un tel outil ressemble à celui d'Apple Quartz Composer, avec la différence qu'il utilise des applications pour ses blocs de construction, on va l'appeler "Applications Composer". Ce concept permet de scripter les applications, et simplifie les tâches complexes pour l'utilisateur. Il réduit la signification d'avoir d'outils alternatives pour accomplir le même but à l'intérieur de la même plate-forme. Si on a plusieurs applications de dessin qui font presque la même chose, le compositeur d'applications éclipse ces applications de leurs fonctions initiales comme des unités indépendantes. Un utilisateur simple qui voulait accomplir une tâche va simplement charger un script, fournit le fichier d'entrée et obtient la sortie, sans besoin de voir en détail comment ceci a été fait, et quelles applications ont été utilisées dans la chaîne. Une partie des développeurs pourraient développer des petites applications sans interface et sans signification si utilisées toutes seules, mais qui sont là pour combler l'écart dans le compositeur. On est en train de parler de l'interaction avec les applications en utilisant des outils, sans se limiter à un interface spécifique. C'est une façon de repenser l'interaction dans une plate-forme comme étant multi-périphérique et multi-application, mais avec un seul design d'interaction [Beaudouin-Lafon, 2004].

A.3.5 Documentation interactive

La méthode actuelle pour créer une documentation pour une application graphique, est soit d'écrire des tutoriels textuels accompagnés d'images des boutons où cliquer, des captures d'écrans des fenêtres de l'application ou d'enregistrer des vidéos de l'auteur entrain d'utiliser l'application avec sa narration. Dans les deux cas, l'utilisateur novice doit commuter entre le tutoriel textuel ou sous forme de vidéo, et l'application réelle qu'il utilise. Il doit commuter plusieurs fois et en même temps suivre l'étape courante du tutoriel. L'utilisateur peut facilement perdre la concentration et oublier de l'étape en cours comme décrit par Laput et al. [Laput et al., 2012]. En utilisant le modèle décrit avec un bus logiciel, la documentation peut être générée indépendamment de l'interface en cours visible à l'utilisateur, mais plutôt en relation avec les fonctionnalités principales de l'application.

Nous avons une nouvelle façon de créer la documentation avec DBus. Par exemple, on peut invoquer une action sur l'application, et on peut aussi détecter ce que l'utilisateur est en train de faire et l'étape courante qu'il est entrain de suivre en utilisant les signaux DBus. Ils peuvent être utilisés pour afficher une partie du tutoriel et ne montrer l'étape suivante que si l'utilisateur a fini ce qui précède. En utilisant les fenêtres transparentes, l'utilisateur ne sera plus obligé de commuter entre l'application qu'il est en train d'apprendre à utiliser et le tutoriel. Ce dernier sera affiché au dessus de l'application même, étape par étape. Si c'est une application de dessin, le tutoriel peut même être écrit dans cette application dans le document courant. En utilisant cette méthode, on a supprimé la commutation entre une application et son tutoriel. Les anciennes formes de documentation (textuels, vidéos) peut être générés avec un script qui appelle automatiquement un logiciel de capture d'écran. Et il va dans ce cas prendre des captures de l'interface actuel selon la plate-forme.

A.4 Conclusion et limites

Dans ce chapitre, nous avons présenté plusieurs scénarios montrant comment la visibilité des méthodes internes d'une application peut être bénéfique pour des nouvelles possibilités de design de plate-formes logicielles. Utilisant ce même concept, nous avons proposé des solutions pour des problèmes comme l'inconsistance dans l'interface et les interactions dans les applications d'une seule plate-forme. Les solutions visent aussi la réduction du temps de développement, le déploiement sur plusieurs périphériques (téléphone, tablette, télévision, ordinateur, ...) et l'unification du système de documentation. Un effet secondaire de notre approche est la réductibilité des applications graphiques créatives envers leurs simple canevas de visualisation. Nous avons expliqué comment ceci apporte une meilleure interaction avec les ordinateurs pour un développeur qui va écrire moins de code, ainsi que pour l'utilisateur qui va avoir sa tâche accomplie en écrivant des courts scripts. Les développeurs ne veulent pas que les outils qu'ils ont développés soient éclipsés et cachés derrière un compositeur d'applications faisant partie de la plate-forme logicielle. Ceci mène aussi à l'utilisation de leurs applications dans des nouveaux cas d'utilisation, largement non intentionnelles, qui nécessite des travaux de standardisation comme ce qui a été fait les lecteurs vidéos sous Linux appelé MPRIS.

Annexe B

Publications

1. Boulabiar, Mohamed-Ikbel, Thomas Burger, Franck Poirier, Gilles COPPIN. A Low-Cost Natural User Interaction Based on Hand Gesture Recognition for Interactive TV. HCI International 2011, Dec 2010, Orlando (FL), United States. pp.214-221
2. Chatty, Stéphane., Boulabiar, Mohamed-Ikbel, and Tissoires, Benjamin. "L'évolution de linux vers les nouvelles formes d'ordinateurs personnels." In Proceedings of the 6th International Conference SETIT 2012, Tunisia, Sousse.
3. Boulabiar, Mohamed-Ikbel, Gilles Coppin, and Franck Poirier. "The Issues of 3D Hand Gesture and Posture Recognition Using the Kinect." Human-Computer Interaction. Springer International Publishing, 2014. 205-214.
4. Boulabiar, Mohamed-Ikbel, Gilles Coppin, and Franck Poirier. "The Study of the Full Cycle of Gesture Interaction, The Continuum between 2D and 3D." Human-Computer Interaction. Springer International Publishing, 2014. 24-35.
5. Boulabiar, Mohamed-Ikbel, Gilles Coppin, and Franck Poirier. "Mashup Architecture for Connecting Graphical Linux Applications Using a Software Bus" Interacción 2014, At Puerto de la Cruz. Tenerife. Spain, 2014.

Bibliographie

[Aglioti et al., 1995] Aglioti, S., DeSouza, J. F., and Goodale, M. a. (1995). Size-contrast illusions deceive the eye but not the hand. *Current biology : CB*, 5(6) :679–85.

[Andric et al., 2013] Andric, M., Solodkin, A., Buccino, G., Goldin-Meadow, S., Rizzolatti, G., and Small, S. L. (2013). Brain function overlaps when people observe emblems, speech, and grasping. *Neuropsychologia*, 51(8) :1619–1629.

[Anthony and Wobbrock, 2010] Anthony, L. and Wobbrock, J. O. (2010). A lightweight multistroke recognizer for user interface prototypes. In *Proceedings of Graphics Interface 2010*, GI '10, pages 245–252, Toronto, Ont., Canada, Canada. Canadian Information Processing Society.

[Baglioni et al., 2009] Baglioni, M., Lecolinet, E., and Guiard, Y. (2009). Espace de caractérisation des interactions gestuelles physiques sur dispositifs mobiles. In *Proceedings of the 21st International Conference on Association Francophone d'Interaction Homme-Machine*, pages 203–212. ACM.

[Bau and Mackay, 2008] Bau, O. and Mackay, W. E. (2008). Octopocus : A dynamic guide for learning gesture-based command sets. In *Proceedings of the 21st Annual ACM Symposium on User Interface Software and Technology*, UIST '08, pages 37–46, New York, NY, USA. ACM.

[Baudel and Beaudouin-Lafon, 1993] Baudel, T. and Beaudouin-Lafon, M. (1993). Charade : remote control of objects using free-hand gestures. *Communications of the ACM*, 36(7) :28–35.

[Beaudouin-Lafon, 2000] Beaudouin-Lafon, M. (2000). Instrumental interaction : an interaction model for designing post-wimp user interfaces. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 446–453. ACM.

[Beaudouin-Lafon, 2004] Beaudouin-Lafon, M. (2004). Designing interaction, not interfaces. In *Proceedings of the Working Conference on Advanced Visual Interfaces*, AVI '04, pages 15–22, New York, NY, USA. ACM.

[Bier et al., 1993] Bier, E. A., Stone, M. C., Pier, K., Buxton, W., and DeRose, T. D. (1993). Toolglass and magic lenses : The see-through interface. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '93, pages 73–80, New York, NY, USA. ACM.

[Boulabiar et al., 2011] Boulabiar, M.-I., Burger, T., Poirier, F., and Coppin, G. (2011). A low-cost natural user interaction based on a camera hand-gestures recognizer. In *Proceedings of the 14th International Conference on Human-computer Interaction : Interaction Techniques and Environments - Volume Part II*, HCII'11, pages 214–221, Berlin, Heidelberg. Springer-Verlag.

[Breuker, 2013] Breuker, J. (2013). A cognitive science perspective on knowledge acquisition. *International Journal of Human-Computer Studies*, 71(2) :177–183.

[Brouet et al., 2013] Brouet, R., Blanch, R., and Cani, M.-P. (2013). Understanding hand degrees of freedom and natural gestures for 3d interaction on tabletop. In *Human-Computer Interaction–INTERACT 2013*, pages 297–314. Springer.

[Bruno and Bernardis, 2002] Bruno, N. and Bernardis, P. (2002). Dissociating perception and action in Kanizsa's compression illusion. *Psychonomic bulletin & review*, 9(4) :723–730.

[Buisson et al., 2002] Buisson, M., Bustico, A., Chatty, S., Colin, F.-R., Jestin, Y., Maury, S., Mertz, C., and Truillet, P. (2002). Ivy : Un bus logiciel au service du développement de prototypes de systèmes interactifs. In *Proceedings of the 14th French-speaking Conference on Human-computer Interaction (Conférence Francophone Sur L'Interaction Homme-Machine)*, IHM '02, pages 223–226, New York, NY, USA. ACM.

[Bullock and Dollar, 2011] Bullock, I. M. and Dollar, A. M. (2011). Classifying human manipulation behavior. In *Rehabilitation Robotics (ICORR), 2011 IEEE International Conference on*, pages 1–6. IEEE.

[Bullock et al., 2012] Bullock, I. M., Ma, R. R., and Dollar, A. M. (2012). A hand-centric classification of human and robot dexterous manipulation. *IEEE Transactions on Haptics*, 6(2) :129–144.

[Cadoz, 1994] Cadoz, C. (1994). Le geste canal de communication homme/machine : la communication'instrumentale'. *Technique et science informatiques*, 13(1) :31–61.

[Casiez et al., 2012] Casiez, G., Roussel, N., and Vogel, D. (2012). 1 euro filter : A simple speed-based low-pass filter for noisy input in interactive systems. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, pages 2527–2530, New York, NY, USA. ACM.

[Chang et al., 2004] Chang, F., Chen, C.-J., and Lu, C.-J. (2004). A linear-time component-labeling algorithm using contour tracing technique. *Comput. Vis. Image Underst.*, 93(2) :206–220.

[Chatty et al., 2007] Chatty, S., Lemort, A., and Vales, S. (2007). Multiple input support in a model-based interaction framework. In *Horizontal*

Interactive Human-Computer Systems, 2007. TABLETOP '07. Second Annual IEEE International Workshop on, pages 179–186.

[Choi et al., 2014] Choi, E., Kim, H., and Chung, M. K. (2014). A taxonomy and notation method for three-dimensional hand gestures. *International Journal of Industrial Ergonomics*, 44(1) :171–188.

[Cutkosky, 1989] Cutkosky, M. (1989). On grasp choice, grasp models, and the design of hands for manufacturing tasks. *Robotics and Automation, IEEE Transactions on*, 5(3) :269–279.

[Dragicevic and Fekete, 2004] Dragicevic, P. and Fekete, J.-D. (2004). Support for input adaptability in the icon toolkit. In *Proceedings of the 6th International Conference on Multimodal Interfaces*, ICMI '04, pages 212–219, New York, NY, USA. ACM.

[Ekman and Friesen, 1981] Ekman, P. and Friesen, W. V. (1981). The repertoire of nonverbal behavior : Categories, origins, usage, and coding. *Nonverbal communication, interaction, and gesture*, pages 57–106.

[Feix et al., 2009] Feix, T., Pawlik, R., Schmiedmayer, H.-b., Romero, J., and Kragic, D. (2009). A comprehensive grasp taxonomy. In *Robotics, Science and Systems : Workshop on Understanding the Human Hand for Advancing Robotic Manipulation*, pages 2–3.

[Gandy et al., 2000] Gandy, M., Starner, T., Auxier, J., and Ashbrook, D. (2000). The gesture pendant : A self-illuminating, wearable, infrared computer vision system for home automation control and medical monitoring. In *Proceedings of the 4th IEEE International Symposium on Wearable Computers*, ISWC '00, pages 87–, Washington, DC, USA. IEEE Computer Society.

[Goldstein, 2008] Goldstein, E. B. (2008). *The Blackwell Handbook of Sensation and Perception*. John Wiley & Sons.

[Goodale and Milner, 1992] Goodale, M. A. and Milner, A. D. (1992). Separate visual pathways for perception and action. *Trends in neurosciences*, 15(1) :20–25.

[Groh and Sparks, 1996] Groh, J. M. and Sparks, D. L. (1996). Saccades to somatosensory targets. i. behavioral characteristics. *Journal of Neurophysiology*, 75(1) :412–427.

[Gupta et al., 2012] Gupta, S., Morris, D., Patel, S., and Tan, D. (2012). Soundwave : using the doppler effect to sense gestures. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1911–1914. ACM.

[Haggard and Flanagan, 1996] Haggard, P. and Flanagan, J. R. (1996). *Hand and brain : the neurophysiology and psychology of hand movements*. Academic Press.

[Harrison and Hudson, 2008] Harrison, C. and Hudson, S. E. (2008). Scratch input : creating large, inexpensive, unpowered and mobile finger input surfaces. In *Proceedings of the 21st annual ACM symposium on User interface software and technology*, pages 205–208. ACM.

[Harrison et al., 2011] Harrison, C., Schwarz, J., and Hudson, S. E. (2011). TapSense : enhancing finger interaction on touch surfaces. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pages 627–636. ACM.

[Harrison et al., 2014] Harrison, C., Xiao, R., Schwarz, J., and Hudson, S. E. (2014). Touchtools : leveraging familiarity and skill with physical tools to augment touch interaction. In *Proceedings of the 32nd annual ACM conference on Human factors in computing systems*, pages 2913–2916. ACM.

[Hauk et al., 2004] Hauk, O., Johnsrude, I., and Pulvermüller, F. (2004). Somatotopic representation of action words in human motor and premotor cortex. *Neuron*, 41(2) :301–307.

[Hilliges et al., 2009] Hilliges, O., Izadi, S., Wilson, A., Hodges, S., Garcia-Mendoza, A., and Butz, A. (2009). Interactions in the air : adding further depth to interactive tabletops. In *Proceedings of the 22nd annual ACM symposium on User interface software and technology*, pages 139–148. ACM.

[Hinckley et al., 1998] Hinckley, K., Pausch, R., Proffitt, D., and Kassell, N. F. (1998). Two-handed virtual manipulation. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 5(3) :260–302.

[Jacob et al., 1994] Jacob, R. J., Sibert, L. E., McFarlane, D. C., and Mullen Jr, M. P. (1994). Integrality and separability of input devices. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 1(1) :3–26.

[Kaltenbrunner et al., 2005] Kaltenbrunner, M., Bovermann, T., Bencina, R., and Costanza, E. (2005). Tuio : A protocol for table-top tangible user interfaces. In *Proc. of the The 6th Int'l Workshop on Gesture in Human-Computer Interaction and Simulation*.

[Karam and Schraefel, 2005] Karam, M. and Schraefel, m. c. (2005). A Taxonomy of Gestures in Human Computer Interactions.

[Kendon, 1972] Kendon, A. (1972). Some relationships between body motion and speech. *Studies in dyadic communication*, 7 :177.

[Kin et al., 2012] Kin, K., Hartmann, B., DeRose, T., and Agrawala, M. (2012). Proton : multitouch gestures as regular expressions. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 2885–2894. ACM.

[Kratz and Rohs, 2010] Kratz, S. and Rohs, M. (2010). A $3 gesture recognizer : Simple gesture recognition for devices equipped with 3d acceleration sensors. In *Proceedings of the 15th International Conference on Intelligent User Interfaces*, IUI '10, pages 341–344, New York, NY, USA. ACM.

[Kratz et al., 2013] Kratz, S., Rohs, M., and Essl, G. (2013). Combining acceleration and gyroscope data for motion gesture recognition using classifiers with dimensionality constraints. In *Proceedings of the 2013 International Conference on Intelligent User Interfaces*, IUI '13, pages 173–178, New York, NY, USA. ACM.

[Kurz, 2014] Kurz, D. (2014). Thermal touch : Thermography-enabled everywhere touch interfaces for mobile augmented reality applications. In *Mixed and Augmented Reality (ISMAR), 2014 IEEE International Symposium on*, pages 9–16. IEEE.

[Lackner, 1988] Lackner, J. R. (1988). Some proprioceptive influences on the perceptual representation of body shape and orientation. *Brain*, 111(2) :281–297.

[Laput et al., 2012] Laput, G., Adar, E., Dontcheva, M., and Li, W. (2012). Tutorial-based interfaces for cloud-enabled applications. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology*, UIST '12, pages 113–122, New York, NY, USA. ACM.

[Larson et al., 2011] Larson, E., Cohn, G., Gupta, S., Ren, X., Harrison, B., Fox, D., and Patel, S. (2011). Heatwave : thermal imaging for surface user interaction. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 2565–2574. ACM.

[Li, 2010] Li, Y. (2010). Protractor : A fast and accurate gesture recognizer. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '10, pages 2169–2172, New York, NY, USA. ACM.

[Lucas, 1985] Lucas, B. D. (1985). *Generalized image matching by the method of differences*. PhD thesis, Carnegie Mellon University.

[Luria et al., 1978] Luria, A., Heissler, N., and Semenov-Ségur, G. (1978). *Les fonctions corticales supérieures de l'homme*. Psychologie d'aujourd'hui. Presses Universitaires de France.

[MacKenzie and Iberall, 1994] MacKenzie, C. L. C. and Iberall, T. (1994). *The grasping hand*. Elsevier.

[Mackinlay et al., 1990] Mackinlay, J., Card, S. K., and Robertson, G. G. (1990). A semantic analysis of the design space of input devices. *Human-Computer Interaction*, 5(2) :145–190.

[Marquardt et al., 2011] Marquardt, N., Jota, R., Greenberg, S., and Jorge, J. A. (2011). The continuous interaction space : Interaction techniques unifying touch and gesture on and above a digital surface. In *Proceedings of the 13th IFIP TC 13 International Conference on Human-computer Interaction - Volume Part III*, INTERACT'11, pages 461–476, Berlin, Heidelberg. Springer-Verlag.

[Martinet et al., 2010] Martinet, A., Casiez, G., and Grisoni, L. (2010). The design and evaluation of 3d positioning techniques for multi-touch displays. In *3D User Interfaces (3DUI), 2010 IEEE Symposium on*, pages 115–118. IEEE.

[McNeill, 2005] McNeill, D. (2005). *Gesture and Thought*. University of Chicago Press.

[Michel et al., 2011] Michel, D., Oikonomidis, I., and Argyros, A. (2011). Scale invariant and deformation tolerant partial shape matching. *Image and Vision Computing*, 29(7) :459–469.

[Mine et al., 1997] Mine, M. R., Brooks, J. F. P., and Sequin, C. H. (1997). Moving Objects in Space : Exploiting Proprioception in Virtual-environment Interaction. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '97, pages 19–26, New York, NY, USA. ACM Press/Addison-Wesley Publishing Co.

[Mistry et al., 2009] Mistry, P., Maes, P., and Chang, L. (2009). Wuw-wear ur world : a wearable gestural interface. In *CHI'09 extended abstracts on Human factors in computing systems*, pages 4111–4116. ACM.

[Mitra and Acharya, 2007] Mitra, S. and Acharya, T. (2007). Gesture recognition : A survey. *Systems, Man, and Cybernetics, Part C : Applications and Reviews, IEEE Transactions on*, 37(3) :311–324.

[Napier and Tuttle, 1993] Napier, J. and Tuttle, R. (1993). *Hands*. Natural science. Princeton University Press.

[Napier, 1956] Napier, J. J. (1956). The prehensile movements of the human hand. *Surger*, 38(4) :902–913.

[Norman, 2002] Norman, D. A. (2002). *The design of everyday things*.

[Oikonomidis et al., 2011a] Oikonomidis, I., Kyriazis, N., and Argyros, A. A. (2011a). Efficient model-based 3d tracking of hand articulations using kinect. In *BMVC*, volume 1, page 3.

[Oikonomidis et al., 2011b] Oikonomidis, I., Kyriazis, N., and Argyros, A. A. (2011b). Full dof tracking of a hand interacting with an object by modeling occlusions and physical constraints. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2088–2095. IEEE.

[Oikonomidis et al., 2012] Oikonomidis, I., Kyriazis, N., and Argyros, A. A. (2012). Tracking the articulated motion of two strongly interacting hands. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 1862–1869. IEEE.

[Olafsdottir et al., 2014] Olafsdottir, H. B., Tsandilas, T., and Appert, C. (2014). Prospective motor control on tabletops : planning grasp for multitouch interaction. In *Proceedings of the 32nd annual ACM conference on Human factors in computing systems*, pages 2893–2902. ACM.

[Olivo et al., 2011] Olivo, P., Marchal, D., and Roussel, N. (2011). Software requirements for a (more) manageable multi-touch ecosystem. In *EICS 2011 Workshop on Engineering Patterns for Multi-Touch Interfaces*.

[Olsen et al., 2007] Olsen, L., Samavati, F. F., and Sousa, M. C. (2007). Fast Stroke Matching by Angle Quantization. *Proceedings of the ImmersCom*.

[Pavlovic et al., 1997] Pavlovic, V. I., Sharma, R., and Huang, T. S. (1997). Visual interpretation of hand gestures for human-computer interaction : A review. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(7) :677–695.

[Quek et al., 2002] Quek, F., McNeill, D., Bryll, R., Duncan, S., Ma, X.-F., Kirbas, C., McCullough, K. E., and Ansari, R. (2002). Multimodal human discourse : gesture and speech. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 9(3) :171–193.

[Reisman et al., 2009] Reisman, J. L., Davidson, P. L., and Han, J. Y. (2009). A screen-space formulation for 2d and 3d direct manipulation. In *Proceedings of the 22nd annual ACM symposium on User interface software and technology*, pages 69–78. ACM.

[Rubine, 1991] Rubine, D. (1991). Specifying gestures by example. *ACM SIGGRAPH Computer Graphics*, 25(4) :329–337.

[Rubine, 1992] Rubine, D. (1992). Combining gestures and direct manipulation. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 659–660. ACM.

[Saba et al., 2012] Saba, E. N., Larson, E. C., and Patel, S. N. (2012). Dante vision : In-air and touch gesture sensing for natural surface interaction with combined depth and thermal cameras. In *Emerging Signal Processing Applications (ESPA), 2012 IEEE International Conference on*, pages 167–170. IEEE.

[Scoditti, 2011] Scoditti, A. (2011). *Gestural interaction techniques for handheld devices combining accelerometers and multipoint touch screens*. PhD thesis, Universite de Grenoble.

[Scoditti et al., 2011] Scoditti, A., Blanch, R., and Coutaz, J. (2011). A novel taxonomy for gestural interaction techniques based on accelerometers. In *Proceedings of the 16th international conference on Intelligent user interfaces*, pages 63–72. ACM.

[Senior et al., 2006] Senior, A., Hampapur, A., Tian, Y.-L., Brown, L., Pankanti, S., and Bolle, R. (2006). Appearance models for occlusion handling. *Image and Vision Computing*, 24(11) :1233 – 1243. Performance Evaluation of Tracking and Surveillance.

[Sève-Ferrieu, 2005] Sève-Ferrieu, N. (2005). *Neuropsychologie corporelle, visuelle et gestuelle : Du trouble à la rééducation*. Elsevier Masson.

[Stuerzlinger et al., 2006] Stuerzlinger, W., Chapuis, O., Phillips, D., and Roussel, N. (2006). User interface façades : Towards fully adaptable user interfaces. In *Proceedings of the 19th Annual ACM Symposium on User Interface Software and Technology*, UIST '06, pages 309–318, New York, NY, USA. ACM.

[Vatavu et al., 2012] Vatavu, R.-D., Anthony, L., and Wobbrock, J. O. (2012). Gestures as point clouds : A $p recognizer for user interface prototypes. In *Proceedings of the 14th ACM International Conference on Multimodal Interaction*, ICMI '12, pages 273–280, New York, NY, USA. ACM.

[Venkataraman et al., 2013] Venkataraman, K., Lelescu, D., Duparré, J., McMahon, A., Molina, G., Chatterjee, P., Mullis, R., and Nayar, S. (2013). Picam : an ultra-thin high performance monolithic camera array. *ACM Transactions on Graphics (TOG)*, 32(6) :166.

[Wadhwa et al., 2013] Wadhwa, N., Rubinstein, M., Durand, F., and Freeman, W. T. (2013). Phase-based video motion processing. *ACM Transactions on Graphics (TOG)*, 32(4) :80.

[Wang and Popović, 2009] Wang, R. Y. and Popović, J. (2009). Real-time hand-tracking with a color glove. In *ACM SIGGRAPH 2009 Papers*, SIGGRAPH '09, pages 63 :1–63 :8, New York, NY, USA. ACM.

[Wilson et al., 2008] Wilson, A., Izadi, S., Hilliges, O., Garcia-Mendoza, A., and Kirk, D. (2008). Bringing physics to the surface. In *Proceedings of the 21st annual ACM symposium on User interface software and technology*, pages 67–76. ACM.

[Wilson, 2009] Wilson, A. D. (2009). Simulating grasping behavior on an imaging interactive surface. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*, pages 125–132. ACM.

[Wimmer, 2011] Wimmer, R. (2011). Grasp sensing for human-computer interaction. In *Proceedings of the fifth international conference on Tangible, embedded, and embodied interaction*, pages 221–228. ACM.

[Wobbrock et al., 2007] Wobbrock, J. O., Wilson, A. D., and Li, Y. (2007). Gestures without libraries, toolkits or training : A $1 recognizer for user interface prototypes. In *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology*, UIST '07, pages 159–168, New York, NY, USA. ACM.

[Wu et al., 2012] Wu, H.-Y., Rubinstein, M., Shih, E., Guttag, J. V., Durand, F., and Freeman, W. T. (2012). Eulerian video magnification for revealing subtle changes in the world. *ACM Trans. Graph.*, 31(4) :65.

[Xiao et al., 2014] Xiao, R., Lew, G., Marsanico, J., Hariharan, D., Hudson, S., and Harrison, C. (2014). Toffee : enabling ad hoc, around-device interaction with acoustic time-of-arrival correlation. In *Proceedings of the 16th international conference on Human-computer interaction with mobile devices & services*, pages 67–76. ACM.

[Xu et al., 2009] Xu, J., Gannon, P. J., Emmorey, K., Smith, J. F., and Braun, A. R. (2009). Symbolic gestures and spoken language are proces-

sed by a common neural system. *Proceedings of the National Academy of Sciences*, 106(49) :20664–20669.

[Yeh et al., 2009] Yeh, T., Chang, T.-H., and Miller, R. C. (2009). Sikuli : Using gui screenshots for search and automation. In *Proceedings of the 22Nd Annual ACM Symposium on User Interface Software and Technology*, UIST '09, pages 183–192, New York, NY, USA. ACM.

[Zeng et al., 2012] Zeng, B., Wang, G., and Lin, X. (2012). A hand gesture based interactive presentation system utilizing heterogeneous cameras. *Tsinghua Science and Technology*, 17(3) :329–336.

Résumé

Dans le domaine de la reconnaissance gestuelle, les mouvements humains sont observés, reconnues et transformés en primitives fonctionnelles pour contrôler un système ou manipuler un objet. Les chercheurs en Interaction Homme-Machine ont plus particulièrement étudié le suivi des gestes réalisés après le contact avec l'objet à manipuler (geste apparent). Nous montrons dans cette thèse qu'interagir et manipuler un objet 3D est un processus plus large qui commence par la vision, qui inclut le rapprochement, la saisie, la manipulation et qui se termine avec la «consommation des événements» dans les applications cibles.

Dans cette thèse, nous avons collecté et organisé un état de l'art sur l'interaction Homme-machine provenant de différents points de vue (vision, neuropsychologie, de saisie et techniques).

Nous avons créé un système qui suit les mouvements des utilisateurs sur une table mais également au dessus de la surface à partir d'un nuage de points 3D. Nous avons spécifié des cas d'activités gestuelles et nous les avons utilisés dans deux applications. Nous avons aussi proposé une nouvelle façon de créer des applications adaptables aux nouvelles formes d'interactions en se basant sur un bus logiciel.

Mots-clés : Geste, Posture, Saisie, Ihm, Cycle, Reconnaissance, 3D, Bus logiciel

Abstract

In the gesture recognition domain, human movements are tracked, recognized and mapped to functional primitives to control a system or to manipulate an object. Human-Computer Interaction researchers have particularly focused on the tracking of the gesture made after the contact with the object to be manipulated (apparent gesture). We show in this thesis that interacting and manipulating a 3D object is a wider process that starts with vision, includes reaching, grasping, manipulating and ends with «event consumption» in target applications.

In this thesis, we have collected and organized a HCI literature review coming from many fields (vision, neuropsychology, grasping and technical).

We have created a system that tracks, from low level 3D point-cloud input, user movements on the table but also above the surface. We have specified multiple cases of gesture activities and we used them in two applications. We have also proposed a novel way of creating future adaptable applications to new forms of interactions using a software bus.

Keywords : Gesture, Posture, Grasping, Hci, Cycle, Recognition, 3D, Software bus

TELECOM Bretagne

n° d'ordre : 2015telb0355
Télécom Bretagne
Technopôle Brest-Iroise - CS 83818 - 29238 Brest Cedex 3
Tél : + 33(0) 29 00 11 11 - Fax : + 33(0) 29 00 10 00